Improving Digital Control System Performance Through a Novel Jitter Compensating Method

submitted by

Chamira Perera, B.Eng

A thesis submitted to
the Faculty of Graduate and Postdoctoral Affairs
in partial fulfilment of
the requirements for the degree of

Master of Applied Science

Ottawa-Carleton Institute for Electrical and Computer Engineering
Faculty of Engineering and Design
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario, Canada K1S 5B6
November, 2013

Copyright © 2013
Chamira Perera

Abstract

SOFTWARE induced delays degrade the performance of a digital control system which can result in violation of performance requirements. Shortcomings of the state-of-the-art include, not considering satisfying plant response characterizing performance requirements, implementation complexity, and extra overhead for the solution. Additionally, a proportional integral derivative (PID) controller solution to compensate for input-output and output jitter, to the best of our knowledge was not found.

This research proposes a jitter compensating PID controller. Firstly, the worst case sampling to output delay, which is one sampling period, is modeled. Secondly, this delay placed between the controller and plant. Finally, the PID controller is designed to satisfy performance requirements. Advantages of this solution include satisfying performance requirements, and unchanged implementation complexity and execution overhead.

Performance results validate the effectiveness of the proposed solution by demonstrating its ability to meet performance requirements in response to a step function, and by tracking a square wave.

Acknowledgments

I would like to take this opportunity to express my gratitude to my supervisor Prof. Trevor Pearce and co-supervisor Prof. Miodrag Bolic for their support not only during my Masters thesis research but also during the entire duration of the Masters program at the Ottawa-Carleton Institute for Electrical and Computer Engineering. The encouragement, guidance, ideas, and advice from both of my supervisors have helped to shape the research into what it is now today.

The assistance from Dr. Anton Cervin affiliated with Lund University, Sweden is also greatly appreciated as it helped me in setting up and becoming familiar with the TrueTime simulator. Whenever I had questions Dr. Cervin always took the time to respond back to me via email. Support from Dr. Mojtaba Ahmadi affiliated with the Mechanical and Aerospace department at Carleton University was valuable when I was trying to understand control systems concepts.

A special heart felt gratitude is expressed to my wife Dr. Hashanie Perera who gave me tremendous amount of encouragement, love, and support during my Masters research.

A final appreciation is directed towards my parents Dr. Sunil Perera and Dr. Ajantha Perera for their support during my Masters research.

Table of Contents

Abstract	ii
Acknowledgments	iii
Table of Contents	iv
List of Figures	viii
List of Tables	ix
List of Abbreviations	xii
List of Terms	xiv
List of Symbols	xvii
 Chapter 1 Introduction	 1
1.1 Thesis Overview	3
1.2 Thesis Contributions	6
 Chapter 2 Background	 7
2.1 Digital Control Systems	7
2.1.1 Overview	7
2.1.2 PID Controllers	9
2.1.3 LQG and State Feedback Controllers	10

2.1.4	Networked Digital Control Systems	11
2.1.5	Implementation Considerations	11
2.1.6	Measuring Performance	16
2.2	Performance Issues in Digital Control Systems	17
2.3	Software Used for Design and Simulation	19
2.3.1	MATLAB and Simulink	20
2.3.2	TrueTime	21
2.4	Example Plants Used for Performance Evaluation	22
2.4.1	Hard Disk Servo	22
2.4.2	Voltage Stabilizer	24
2.4.3	An Industrial Process	25
 Chapter 3 State-of-the-Art in Improving Performance of Digital		
	Control Systems Due to Software Induced Delays	27
3.1	Compensating Delays in the Digital Controller	28
3.1.1	Compensating for a Constant Delay	28
3.1.2	Compensating for Jitter	30
3.2	Reducing Delays	33
3.2.1	Software Based Solutions	34
3.2.1.1	Relative Deadline Assignment	34
3.2.1.2	Task Splitting	35
3.2.1.3	Changing the Scheduling Algorithm	35
3.2.1.4	Dynamic Voltage Scaling	36
3.2.1.5	Comparison of Solutions Types	36
3.2.2	Hardware Based Solutions	38
3.3	Compensating for Delays Combined with Delay Reduction	38
 Chapter 4 A Novel Solution to Improve Performance of Digital		
	Control Systems Due to Software Induced Jitter	41

4.1	Analysis of the State-of-the-Art	42
4.2	The Thesis	46
4.3	Thesis Scope	46
4.4	Proving the Thesis Claim	47
Chapter 5	Modelling the System	49
5.1	System Composition	50
5.2	Timing Model	51
5.3	System Configurations	53
5.3.1	Configuration 1: Hardware Input and Hardware Output Con- figuration	54
5.3.2	Configuration 2: Hardware Input and Software Output Config- uration	56
5.3.3	Configuration 3: Software Input and Hardware Output Config- uration	58
5.3.4	Configuration 4: Software Input and Software Output Config- uration	60
5.3.5	Comparison of Configurations	62
Chapter 6	Jitter Compensating PID Controller Solution	63
6.1	Design Overview	63
6.2	Continuous-Time Design	64
6.3	Discrete-Time Design	67
6.4	Summary of the Design Process	68
6.5	Improving TrueTime	69
Chapter 7	Performance Evaluation	72
7.1	Performance Requirements	72
7.2	Experimental Setup	73

7.3	Experimental Results	78
7.3.1	Results for the Hard Disk Servo Digital Control System	79
7.3.2	Results for the Voltage Stabilizer Digital Control System	84
7.3.3	Results for the Industrial Process Digital Control System	88
7.4	General Discussion of the Experimental Results	93
7.5	Limitations of the Proposed Solution	95
Chapter 8	Conclusions and Future Research	100
8.1	Summary of Contributions	103
8.2	Recommendations for Future Research	103
References		106

List of Figures

Figure 2.1:	Typical Model of a Digital Control System	8
Figure 2.2:	Timing Diagram of a Typical Digital Control System	14
Figure 2.3:	Simulink PID Controller Tuner Interface	21
Figure 2.4:	Hard Disk Servo Plant	23
Figure 2.5:	Voltage Stabilizer Plant	24
Figure 5.1:	System Model of a Digital Control System	51
Figure 5.2:	Timing Diagram of Configuration 1	55
Figure 5.3:	Timing Diagram of Configuration 2	57
Figure 5.4:	Timing Diagram of Configuration 3	59
Figure 5.5:	Timing Diagram of Configuration 4	61
Figure 6.1:	Continuous-Time Control System Model with the Delay Element	66
Figure 6.2:	Discrete-Time Control System with a Delay	67
Figure 7.1:	Simulink Simulation Model	78

List of Tables

Table 3.1:	Comparison of Software Based Solutions That Reduce Delays and Jitter	37
Table 5.1:	Summary of Configuration 1 Timing Parameters	55
Table 5.2:	Summary of Configuration 2 Timing Parameters	57
Table 5.3:	Summary of Configuration 3 Timing Parameters	60
Table 5.4:	Summary of Configuration 4 Timing Parameters	61
Table 5.5:	Comparison of the Four Configurations	62
Table 7.1:	Design Parameters for Each Digital Control System Example . . .	73
Table 7.2:	Control Task Periods Used in Each Digital Control System Set . .	75
Table 7.3:	Processor Utilizations for Each Configuration	77
Table 7.4:	Hard Disk Servo Digital Control System Average Step Input Performance	80
Table 7.5:	Hard Disk Servo Digital Control System Step Input Performance 95% Confidence Intervals	81
Table 7.6:	Step Input Relative Performance Data of System Types Which Use the Hard Disk Servo Digital Control System	82
Table 7.7:	Hard Disk Servo Digital Control System Average Tracking Performance	83
Table 7.8:	Hard Disk Servo Digital Control System Tracking Performance 95% Confidence Intervals	83

Table 7.9: Tracking Relative Performance of System Types Which use the Hard Disk Servo Digital Control System	84
Table 7.10: Voltage Stabilizer Digital Control System Average Step Input Per- formance	84
Table 7.11: Voltage Stabilizer Digital Control System Step Input Performance 95% Confidence Intervals	85
Table 7.12: Step Input Relative Performance Data of System Types Which Use the Voltage Stabilizer Digital Control System	86
Table 7.13: Voltage Stabilizer Digital Control System Average Tracking Perfor- mance	87
Table 7.14: Voltage Stabilizer Digital Control System Tracking Performance 95% Confidence Intervals	87
Table 7.15: Tracking Relative Performance of System Types Which use the Voltage Stabilizer Digital Control System	88
Table 7.16: Industrial Process Digital Control System Average Step Input Per- formance	89
Table 7.17: Industrial Process Digital Control System Step Input Performance 95% Confidence Intervals	90
Table 7.18: Step Input Relative Performance Data of System Types Which Use the Industrial Process Digital Control System	91
Table 7.19: Industrial Process Digital Control System Average Tracking Perfor- mance	92
Table 7.20: Industrial Process Digital Control System Tracking Performance 95% Confidence Intervals	92
Table 7.21: Tracking Relative Performance of System Types Which use the Industrial Process Digital Control System	92
Table 7.22: Performance Requirements for the Voltage Stabilizer Digital Con- trol System with a Faster Response	97

Table 7.23: Average Step Input Performance for Voltage Stabilizer Digital Control System with a Faster Response	97
Table 7.24: 95% Confidence Intervals for Step Input Performance for Voltage Stabilizer Digital Control System with a Faster Response	98

List of Abbreviations

A/D	Analog to Digital
ADC	Analog to Digital Converter
API	Application Programming Interface
COUS	Control Output Update State
CS	Context Switch
D/A	Digital to Analog
DAC	Digital to Analog Converter
DCTS	Distance-Constrained Task System
DM	Deadline Monotonic
DVS	Dynamic Voltage Scaling
EDF	Earliest Deadline First
IAE	Integrated Absolute Error
IMF	Initial Mandatory Final
ISE	Integrated Squared Error
ISR	Interrupt Service Routine
ITAE	Integrated Time Absolute Error
LQG	Linear Quadratic Gaussian
MIMO	Multiple Input Multiple Output
MPC	Model Predictive Control
PD	Proportional Derivative
PI	Proportional Integral

PID	Proportional Integral Derivative
RM	Rate Monotonic
RTOS	Real-Time Operating System
SISO	Single Input Single Output
VCM	Voice Coil Motor

List of Terms

Computational delay	Sum of delays in the control task resulting from the execution of software and hardware (A/D, D/A conversions) without any interference from other control tasks; the computational delay can vary as a result of the implementation of the A/D and D/A conversions, control law calculation, ISRs, and RTOS scheduler.
Configuration	Represents a realistic way of implementing a digital control system, where software and/or hardware is used to trigger the two ADCs and the DAC; this results in four distinct configurations.
Conventional solution	An existing solution in the literature that compensates for input-output jitter; this solutions typically uses a prediction based technique for the compensation.
Digital control system set	A collection of independently executing SISO digital control systems; all digital controllers execute simultaneously on the same processor.

Input-output jitter	Variation in the time to output the control law to the plant from the start of sampling inputs from one job to the other for a control task.
One sampling period delay assumption	The assumption that the worst case response time of a job for a control task is equal to one sampling period.
Output jitter	Variation of the time in outputting the control law to the plant between successive jobs for a particular control task.
PerfCost	Estimating performance using a cost function and comparison between a reference and/or other solutions in the literature.
PerfPlantReqs	Measuring performance using plant response characterizing metrics and formulating requirements based on these metrics; a cost function can also be used to estimate performance.
PerfQual	Qualitative assessment of the plant response graphically without quantifying performance.
PerfSW	Measuring performance using software related performance metrics such as jitter, occasionally, a cost function is also used.
Sampling interval	The time between sampling the inputs for successive jobs for a particular control task when sampling jitter is present; when sampling jitter is present the sampling interval is not equal to the sampling period.

Sampling jitter	Variation of time in sampling the inputs between successive jobs for a particular control task.
Sampling period	The time between successive input sampling when sampling jitter is not present.

List of Symbols

Greek Symbols

$\tau_{i,j}$ Computational delay for job $J_{i,j}$ of control task T_i .

Roman Symbols

$\dot{h}_{i,j}$ Sampling interval between successive jobs ($J_{i,j-1}$ and $J_{i,j}$) of control task T_i .

h_i Sampling period of control task T_i .

$InOut_{i,j}$ Time difference between input sampling to outputting the control law for job $J_{i,j}$ of control task T_i .

$J_{i,j}$ A job or an instantiation of control task T_i .

$J_{i,j}^C$ A job of the control law calculation periodic task T_i^C .

$J_{i,j}^I$ A job of the input periodic task T_i^I .

O_i Fixed relative offset from $r_{i,j}$ where a hardware timer expires and the DAC is triggered to output the control law for control task T_i .

$Out_{i,j}$ Time difference between outputting the control signal for jobs $J_{i,j-1}$ and $J_{i,j}$ of control task T_i .

$R_{i,j}$ Response time for job $J_{i,j}$ of control task T_i .

$r_{i,j}$ Release time for job $J_{i,j}$ of control task T_i .

T_i A periodic digital control task.

T_i^C A software periodic task that calculates the PID control law for control

task T_i .

T_i^I A software periodic task that triggers both ADCs for control task T_i .

Chapter 1

Introduction

Advances in computer technology has replaced analog controllers with digital controllers with the promise of increased flexibility in implementing complex control algorithms, reduction in cost, and increased noise immunity [1]. In spite of the advantages, there are challenges in implementing digital control systems. Software induced delays, analog to digital (A/D) quantization and round-off effects, and aliasing of input signals are some of these challenges and a multitude of literature exists to rectify these problems.

An inherent problem with any piece of software is the execution overhead, also known as software induced delay, and this is no exception for digital control systems. However, traditional approaches utilized in designing digital control systems overlook the execution overhead and assume input sampling, control law calculation, and outputting the control signal all happen simultaneously [2]. Therefore, the research described in this dissertation focuses on the problems faced due to software induced delays in digital control systems.

If the workflow of a digital control system is conceptualized using a periodic control task, the delays at the task level include the A/D and digital to analog (D/A) conversions, and variable execution time of branch targets in the control law calculation and real-time operating system (RTOS) execution. These delays can vary from

one period to the other. This variation of delays from one period (also known as sampling period) to the other is known as **jitter**. The delays resulting from the A/D and D/A conversions are hardware induced delays, however, these are also considered in this research. The execution of more than one control task on the same processor increases the software induced delays and as a result the amount of jitter also increases. When multiple software tasks execute on the same processor they compete for resources (processor and memory). This competition interferes with each task's execution.

The effect of any kind of delay, including software induced delay, in a digital control system is performance degradation. As the delay increases, the performance degradation increases in tandem. The undesirable outcome of performance degradation is a violation of performance requirements, which manifests after the delay passes a certain threshold. The performance requirements considered in this research are based on characterizing the plant response which are, rise time, settling time, and percent overshoot. In the worst case, the delay could cause the digital control system to become unstable. The effects of delay in a digital control system, along with background information related to digital control systems and real-time systems pertaining to this research are discussed in Chapter 2.

Prior art solutions have attempted to address the performance degradation issue due to software induced delays and they are classified as: solutions from a control systems perspective which compensates for jitter by the controller, solutions from a real-time systems perspective which modifies how tasks are executed and scheduled to reduce jitter, and finally a combined approach known as control scheduling co-design. Shortcomings identified in the literature include but are not limited to, implementation complexity and prediction errors because of plant modelling errors, and an increase in the execution overhead. Solutions from the real-time systems perspective have an inherent problem where the performance of the control system is not considered. In fact, a common problem with the majority of the solutions from all

perspectives is not prioritizing satisfying performance requirements based on metrics that characterize the response of the plant. These include percent overshoot, rise time, and settling time. All solutions, however, prioritize satisfying stability.

Solutions from the control systems and control scheduling co-design perspectives usually estimate the performance using a cost function and compare it against a reference and/or other solutions in the literature (defined as *PerfCost*). Solutions from the real-time systems perspective measure software related metrics such as jitter and input to output delay, and also compares them against existing solutions (defined as *PerfSW*). Occasionally, solutions employ a qualitative approach to assess performance where the plant response is compared graphically against a reference and/or other solutions, without quantifying the performance (defined as *PerfQual*). This research uses a combination of a cost function and plant response characterizing metrics to assess performance (known as *PerfPlantReqs*). A comprehensive review of the current state-of-the-art solutions to improve jitter related performance is discussed in Chapter 3.

The survey of the literature also shows the lack of a solution to compensate for input-output jitter and output jitter using a proportional integral derivative (PID) controller. The low implementation complexity has made PID controllers one of the widely used controllers in industry. This is exemplified by the fact that PID controllers account for 95% of the controllers used in process control [3]. Because of the wide use of PID controllers, it would be valuable to compensate for jitter and improve performance.

1.1 Thesis Overview

The thesis of the research proposes a PID controller that is capable of compensating for software induced jitter. This research considers a collection of single input single output (SISO) digital control systems executing simultaneously, where all digital

controllers execute on the same processor. This collection of digital control systems is defined as a **digital control system set** in this dissertation. The compensation involves initially modelling the worst case delay between input sampling to outputting of the control signal to the plant. This delay is equal to one sampling period. Conventional approaches found in the literature used for designing input-output jitter and output jitter compensating digital control systems usually require that the control signal be delivered to the plant at a fixed time. The start of the sampling period is typically used as the fixed instant in time. This is because the time between input sampling and outputting the control signal needs to be determined ahead of time. Once this time difference is determined, the control signal which is applied at the fixed time is predicted. The prediction is performed after sampling the inputs. However, the proposed solution is not constrained by outputting the control signal at a fixed time, but instead, it can be applied to some representatives of a range of digital control systems.

The final outcome of this research is a PID controller that has exactly the same execution time as its uncompensated counterpart. Therefore, a digital control system which is constrained by a limited timing budget is immensely benefited by the proposed solution. Chapter 4 introduces the solution presented in the thesis, and highlights the contributions emanating from this research. The scope of the research, and an overview of how the thesis claim is validated are discussed in Chapter 4.

Subsequently, Chapter 5, elaborates on the model of the digital control system set used in this research and its timing properties. The Chapter also presents four realistic ways of implementing digital control systems. Each of these are referred to as a configuration. The main contribution of this research, which is a jitter compensating PID controller, is discussed in detail in Chapter 6.

A simulation based performance evaluation is used to compare a digital control

system set with the proposed jitter compensating PID controller (defined as *NewComp*) against a digital control system set without the jitter compensating PID controller (defined as *UnComp*). The performance of a digital control system set with the proposed jitter compensating PID controller that delivers the control signal at the start of the sampling period (defined as *CompOnePeriod*), akin to a conventional design, is also used in the comparison. Three plant examples found in the literature are used in the performance evaluation.

Two types of experiments are carried out in the performance evaluation: the first experiment evaluates the performance in response to a step function, while the second experiment evaluates the performance related to tracking a square wave. Performance results from the step input experiment illustrate that the *NewComp* and *CompOnePeriod* types satisfied performance requirements. The *UnComp* type was not able to satisfy one or more requirements. The *NewComp* type performed better compared to *UnComp* and *CompOnePeriod*. Nevertheless, an improvement in the rise time was not consistently observed in the *NewComp* type compared to the *UnComp* and *CompOnePeriod* types. The tracking performance results also exemplified the improvement in performance of the *NewComp* compared to *UnComp* and *CompOnePeriod* types. These performance results successfully validate the thesis claim presented in Chapter 4. A detailed treatment of the performance evaluation is provided in Chapter 7, and performance results are also discussed. The Chapter also discusses a limitation of the proposed solution and proposes a workaround.

Finally, Chapter 8, summarizes the research performed to improve the jitter related performance in digital control systems by utilizing a jitter compensating PID controller. The Chapter also proposes ways to improve the research.

1.2 Thesis Contributions

The contributions of this thesis are with respect to improving the performance of a digital control system in the presence of software induced jitter. This Section lists the main contributions made through the thesis.

Contribution 1: A design procedure to obtain a software induced jitter compensating PID controller which does not increase the execution overhead - The main contribution emanating from this research is a PID controller that compensates for software induced jitter. The design of the jitter compensating PID controller relies on the automated PID tuning tool from MATLAB and Simulink. Experimental results demonstrate that the jitter compensating PID controller applied to some representatives of a range of digital control systems that respect the one sampling period delay assumption, is capable of satisfying plant response characterizing performance requirements that an uncompensated PID controller failed to satisfy. Chapter 6 outlines the design procedure to obtain the jitter compensating PID controller.

Contribution 2: Improving the TrueTime simulator - Support was added to the TrueTime simulator to simulate timed hardware triggers. The intent behind this feature is to trigger analog to digital converters (ADCs) and digital to analog converter (DAC) exactly at precise instants in time to eliminate sampling jitter and minimize input-output jitter and output jitter as much as possible. This is a minor contribution compared to the main contribution. Information pertaining to this feature is presented in detail in Chapter 6, Section 6.5.

Chapter 2

Background

Digital control systems are designed to control a plant to achieve a desired response. Delays that arise due to executing software that implements the functionality of the digital control system can degrade the performance of the digital control system. To that end, the research outlined in this dissertation investigates a solution to improve performance of a digital control system implemented on a computer system. This Chapter provides background information on digital control systems, real-time systems, and related performance issues, along with software tools and example plants used in the research.

2.1 Digital Control Systems

The intention of this Section is not to provide a detailed treatment of the subject matter, but instead, information relevant to this research is only highlighted.

2.1.1 Overview

The intent of using a control system is to regulate a plant to match a setpoint (also known as reference input) as closely as possible. The controller connected to the

plant is responsible for comparing the reference input against the output response of the plant and making the necessary adjustments in a feedback loop. This action is referred to as closed loop control. Open loop control also exists and it does not use feedback from the response of the plant to control the plant [4]. Since this type of control is typically used in very simple systems, it is not considered in this research, but instead closed loop control is only considered.

A digital control system also known as a computer controlled system has its controller implemented in a digital computer [1]. Figure 2.1 shows a high-level block diagram of a SISO digital control system and the components implemented in the computer.

Figure 2.1: Typical Model of a Digital Control System

The clock in the digital control system provides a source of timing for all digital components. However, the clock used drive the controller typically operates at a high frequency compared to the clock used for the ADCs and DAC. The workflow for closed loop control in a digital control system consists of:

1. Discretizing the inputs using ADCs; sample and hold, followed by quantization of the the two inputs $r(t)$ and $y(t)$ to produce $r(kh)$ and $y(kh)$.

2. Comparing $r(kh)$ and $y(kh)$ to produce the error signal $e(kh)$.
3. Executing the control law by the controller to produce the control signal $u(kh)$.
4. Converting $u(kh)$ to a continuous time signal $u(t)$ using a DAC and outputting it to the plant.

This workflow repeats indefinitely for every period h and it can be considered to be a periodic task. This periodic task is referred to as the control task in this dissertation. The quantity h is also known as the **sampling period** and it is defined as the time between successive sampling instants. Before computers became commonplace, control systems were implemented exclusively using analog components. Advances in computer technology and the advantages of implementing controllers in computers have made the implementation of controllers using analog components redundant. The advantages of digital control systems over their analog counterparts are: reduction in cost, power, and size, ease of implementing complex control algorithms and accommodating changes, noise immunity, and elimination of parameter drift [1] [5] [6].

2.1.2 PID Controllers

The controller in the control system is instrumental in achieving the desired response from the plant. The PID control strategy is the one of many ways to implement a controller. These controllers represent 95% of the controllers used in process control [3] and are widely used in many industries owing to its simplicity. The discrete-time realization of a PID controller is given by (2.1), [1].

$$u(kh) = P(kh) + I(kh) + D(kh) \quad (2.1)$$

The P - proportional term deals with the present error in the digital control system, and it increases the speed of the plant response. However, it can cause the plant response to have a high overshoot. The I - integral term deals with the past

errors in the digital control system. It is helpful in reducing the steady state error. The D - derivative term deals with the future errors in the digital control system, and its role is to improve the stability of the plant response [7]. On the other hand, the presence of measurement noise in practical systems, along with the derivative term degrades the performance of the digital control system. Therefore, in practice, the derivative term is almost always accompanied with a low pass filter to remove unwanted measurement noise [3]. The derivative term is able to identify whether the error increases or decreases by examining the direction of the tangent [3], [4], [8].

Limitations of PID controllers become apparent when trying to control plants with higher order dynamics (greater than two), large dead-time, and oscillatory modes [3]. In these situations more sophisticated control strategies such as model predictive control (MPC) have to be used.

Typically, the I and D terms are never used in isolation. However, all controllers use the P term. In addition, the I or D terms can be used alone in conjunction with a P term. In certain situations the integral action in a PID controller has to be turned off due to a dominant integrator in the controlled plant. These types of plants with a dominant integrator are defined as type 1 systems in the literature. The resulting controller is known as a proportional derivative (PD) controller. When controlling type 1 systems, a PD controller is indispensable in order to achieve a steady state error of zero [9].

2.1.3 LQG and State Feedback Controllers

Apart from PID controllers, two other popular control strategies used for digital control systems are, linear quadratic Gaussian (LQG) and state feedback control.

The objective of state feedback control is to control a plant by measuring each state variable [10]. However, in practice, measuring all state variables is not feasible. Therefore, an observer is used to estimate the states required by the controller. The observer relies on a model of the plant to estimate the states. This implies, the

accuracy of the state estimation is heavily dependent on the accuracy of the plant model.

The theory of LQG controllers is based upon optimal control. As a consequence, an optimization problem is solved when designing the controller. As with any optimization problem, the design of LQG controllers attempt to optimize a cost function [11]. Usually, states of the plant are measured or observed for LQG control similar to state feedback control. Despite robust control, a disadvantage of this approach is, specialized knowledge of control systems is required to design LQG controllers, unlike in the case of PID controllers.

2.1.4 Networked Digital Control Systems

Networked digital control systems are a special kind of digital control systems where the components of the digital control system are distributed among different nodes interconnected through a wired or wireless network [12]. That is, the sensors that measure the plant response, the computer that executes the controller, and the plant are distributed. Applications of network digital control systems can be found in the automotive and aerospace domains. The nodes communicate among each other by sending packets. This research only considers digital control systems that have all components on a single node and does not consider network digital control systems.

2.1.5 Implementation Considerations

Digital controllers used in many application domains are typically implemented on embedded microcontrollers. A microcontroller either has the required ADCs and DACs or both are interfaced externally to the microcontroller. Nowadays, most implementations of digital control systems have tasks other than the digital controller executing on the same processor. This fact is exemplified in [13], which presents a robot that executes motor control, artificial intelligence, a wireless stack, and many

signal processing functions on a single processor.

Execution of more than one task on a processor implies that these tasks will be competing for resources and have to be scheduled and also meet timing constraints. Therefore, it would make sense to use a RTOS which implements, for example, the earliest deadline first (EDF) or rate monotonic (RM) priority based preemptive scheduling policies, to guarantee satisfying timing constraints [14]. The priority assignment is fixed for RM, and it is ordered based on the task periods. For example, the task with the shortest period is assigned the highest priority and vice versa. Conversely, the priority assignment for EDF is dynamic. The tasks are assigned priorities inversely proportional to their relative deadlines. If the total processor utilization does not exceed the required utilization bounds for each policy, then the tasks will always be schedulable. In the case of EDF, the utilization bound should not exceed 1, while for RM, the utilization bound is given by (2.2),

$$\sum_{i=1}^n U_i < n (2^{1/n} - 1) \quad (2.2)$$

U_i is the processor utilization for task i , and n is the total amount of tasks considered. The schedulability test given by (2.2) is a sufficient condition. However, in the event that the total processor utilization exceeds the utilization bound, a conclusion cannot be drawn about the schedulability of the tasks. In this case more sophisticated tests such as, time-demand and response time analysis, have to be employed.

The deadline monotonic (DM) scheduling policy uses a fixed priority assignment scheme similar to RM. Additionally, DM is exactly the same as RM when the relative deadline of the tasks are equal to the period [15]. This research focuses on EDF since it has better schedulability of tasks over RM and DM.

This research considers systems with more than one control task executing on a single processor. However, the scope is limited to SISO digital control systems as opposed to multiple input multiple output (MIMO) digital control systems which also

have more than one workflow. The workflows in a MIMO digital control system are dependent on one another and as a result, more complex to implement. Therefore, the systems that this research consider are multiple independently executing SISO digital control systems. In this dissertation the term **digital control system set** refers to a collection of SISO digital control systems, and the terms SISO digital control system and digital control system are synonymous.

To ensure that the processor executing multiple control tasks do not over utilize the processor it is important to chose an appropriate sampling period. An equation to derive the sampling period is given by (2.3) [1].

$$h = \frac{T_r}{N} \quad (2.3)$$

T_r is the required rise time of the plant response. The value for N suggested by Åström et al. is 4 to 10 [1] while J.Liu suggests using 10 to 20 [16]. Equation (2.3) is applied in Chapter 6 and the value of 10 is used as the common value between the two approaches.

The periodic nature of the control task in a digital control system mentioned in Section 2.1.1 implies that the control tasks in the digital control system set can be viewed according to a periodic task model. If the RTOS contains a timer facility with support from the underlying hardware this can be used to schedule the periodic execution of the control task. Triggering the ADCs to sample the inputs, calculating the control law, and triggering the DAC are the only components of the control task executed by software. The software also has to execute the RTOS scheduler and service interrupts. On the other hand, the A/D and D/A conversions are executed in hardware, while the execution of the plant occurs outside of the computer.

Leveraging ideas from Liu's periodic task model [16], the periodic task model in this research considers a digital control system set implemented as a periodic task set $T_i \in \{T_1, T_2, \dots, T_n\}$. T_i corresponds to an individual control task, and it has an

infinite job set $J_{i,j} \in \{J_{i,1}, J_{i,2}, \dots\}$. Job $J_{i,j}$ is an instantiation of a control task T_i . The relative deadline D_i of each control task T_i is equal to its sampling period h_i . Furthermore, the relative deadline $D_{i,j}$ for job $J_{i,j}$ is measured from its release time $r_{i,j}$. The response time $R_{i,j}$ is the time it takes until the control law is output from the release time $r_{i,j}$ for job $J_{i,j}$.

Figure 2.2 depicts a timing diagram of control task T_i for four consecutive jobs $J_{i,k-2}$ to $J_{i,k+1}$. $InOut_{i,j}$ is the time to output the control law by the DAC relative to the start of sampling the inputs for job $J_{i,j}$ (time between events (2) and (3) in Figure 2.2). $Out_{i,j}$ is the time between outputting the control law (event (3) in Figure 2.2) for jobs $J_{i,j-1}$ and $J_{i,j}$. $\hat{h}_{i,j}$ is the time between sampling points (event (2) in Figure 2.2) for jobs $J_{i,j-1}$ and $J_{i,j}$. This is defined as the **sampling interval** for job $J_{i,j}$.

Figure 2.2: Timing Diagram of a Typical Digital Control System

Traditional approaches used in designing digital control systems assume that events (1), (2), and (3) in Figure 2.2 occur simultaneously. Furthermore, these approaches assume events (2) and (3) occur in a periodic manner. However, in a realistic implementation all three events can be *delayed* from their original timing instant. For example, event (3) is delayed by $InOut_{i,j}$ relative to event (2).

Because a digital control system has periodic events, the timing of these events

can vary from one period to the other. This phenomenon is defined as **jitter**.

The values of Out_i are within the range $[Out_i^-, Out_i^+]$, $[InOut_i^-, InOut_i^+]$ for $InOut_i$, and $[\acute{h}_i^-, \acute{h}_i^+]$ for $\acute{h}_i$. This dissertation defines the following three jitter terms [17], [18], [19]:

1. **Sampling jitter:** The variation in $\acute{h}_{i,j}$ for T_i .
2. **Input-output jitter:** The variation in $InOut_{i,j}$ for T_i .
3. **Output jitter:** The variation in $Out_{i,j}$ for T_i .

As noted in Figure 2.2 the notion of sampling interval is not the same as the sampling period in the context of executing multiple control tasks on the same processor. The sampling period is always equal to h_i for task T_i . On the other hand, the sampling interval for job $J_{i,j}$ at times can be $0 < \acute{h}_{i,j} \leq h_i$ or at times can be $h_i \leq \acute{h}_{i,j} < 2h_i$.

Another term relevant to this dissertation is **computational delay** or $\tau_{i,j}$ for control task T_i , is defined as the sum of delays in the control task due to the execution of software and hardware (A/D, D/A conversions) without interferences from other control tasks [1]. In other words the computational delay is the delay between input sampling and outputting the control law to the plant. Computational delay itself can vary (jitter) from one job to the other or be constant, which exclusively depends on the underlying implementation. Examples of where jitter in computational delay can arise are due to non-constant execution time of branch targets (basic blocks) in the software and using algorithms like successive approximation to implement ADCs.

Performance issues that arise due to implementing digital controllers in a computer are highlighted in Section 2.2.

2.1.6 Measuring Performance

Standard time domain metrics are utilized to measure the performance of digital control systems in response to a step input, and they are employed both in the academia and industry. These metrics are [5],

1. **Rise time:** The time it takes for the plant response to go from 10% to 90% of its final value.
2. **Settling time:** The time it takes for the plant response to reach and stay within $\pm 2\%$ (defined as the settling threshold) of its final value.
3. **Percent overshoot:** The value of the maximum peak of the plant response expressed as a percentage of the final value.

These three metrics characterize the plant response, and performance requirements can be formulated using these metrics. Alternatively, a cost function can also be used to estimate the performance of a digital control system using a single metric. Therefore, the advantage of using a cost function is, the ease of comparison of different digital control system implementations. Conversely, estimating performance using cost functions alone do not illustrate the fact that performance requirements have been satisfied or not. As a result, it is paramount to use metrics such as rise time, settling time, and percent overshoot. Controllers can be designed by minimizing cost functions. In this situation the design of the controller becomes an optimization problem.

The integrated absolute error (IAE), integrated squared error (ISE), and integrated time absolute error (ITAE) are three commonly used cost functions and they are given by (2.4), (2.5), and (2.6) respectively [3].

$$IAE = \int_0^{\infty} |e(t)| \, dt \quad (2.4)$$

$$ISE = \int_0^{\infty} e(t)^2 \, dt \quad (2.5)$$

$$ITAE = \int_0^{\infty} t |e(t)| \, dt \quad (2.6)$$

A limitation of the ITAE cost function is, time dependent weighting results in de-emphasizing the transient portion of the plant response [20]. Squaring the error in the ISE cost function results in placing a large weight to large errors. This is mostly an issue when trying to design a controller via optimization. This research employs the IAE cost function to estimate performance.

Tracking a periodic function, for example, a square wave or sine wave is another method to evaluate the performance of a digital control system. The purpose of the tracking test is to determine how closely the plant response follows the periodic function used as the reference input. Utilizing a cost function is the preferred method to estimate the tracking related performance, because it allows different implementations to be compared using a single metric. The three plant response characterizing metrics discussed in this Section cannot be used to measure the tracking related performance, as they are not applicable in a tracking related situation.

2.2 Performance Issues in Digital Control Systems

Literature in the field of digital control systems have proposed solutions to many performance issues that emerge in digital control systems. This Section only describes the most prominent performance issues which are:

Issue 1: Software Induced Delays: The negative effect of software induced delays whether it be constant or jitter results in a performance degradation. This situation

can be undesirable if the outcome of the performance degradation is failing to satisfy performance requirements of the digital control system. In the worst case the digital control system can become unstable. Furthermore, the high frequency noise introduced by input-output jitter can cause unnecessary wearing of certain components in a plant [21]. If the jitters present in the digital control system are small enough then they can be ignored, otherwise they have to be taken into account during the design phase [22]. Solutions to improve performance due to software induced delays are presented in detail in Chapter 3.

Issue 2: Aliasing of Input Signals: The phenomenon of aliasing occurs when trying to sample a signal with unwanted high frequency components. From a theoretical point of view aliasing results when a sampled signal has frequency components greater than half the sampling frequency. The effect of aliasing in digital control systems translate to inaccuracies in the calculation of the control law which also results in performance degradation or instability. The best method to overcome this issue is to add an anti-aliasing filter, which is essentially a low pass filter before sampling the inputs to remove unwanted frequency components.

Issue 3: Limitations in Precision: Limitation in precision refers to the finite limit of representing data in a computer. For example, an ADC has a finite precision of 8 to 31 bits [23], [24], which leads to quantization errors. Round-off errors are due to the finite word size of 32 to 64 bits to represent an integer or floating point number in a processor. If quantization and round-off is not appropriately considered in the design phase it could ultimately lead to limit-cycle oscillations in a digital control system which ultimately results in an oscillatory response of the plant [1]. Changing the control law calculation to accommodate for quantization and round-off errors is a solution to mitigate this issue.

Issue 4: Integrator Windup: Practical implementations of plants have nonlinearities like saturation [8]. Examples include, limited speed for a motor and limitation in valve position (the valve position cannot exceed the open or closed position). If the digital control system uses integral action and if the plant reaches saturation, the integral term also increases because normally in this situation the error signal is non-zero. Once the integral term and the resulting control signal reaches a large value it will take a large time for the integral term to reach a normal value. This effect is known as integral windup. Solutions to overcome integral windup is to stop updating the integrator when the plant has saturated or implement an anti-windup algorithm for a controller [25].

Issue 5: Plant Dead-Time: Plant dead-time is defined as the amount of delay the plant takes to respond after receiving the control signal from the controller. The presence of dead-time in the plant usually complicates the design process as these types of plants are difficult to control. The difficulty arises due to the large delay which can be many times larger than the sampling period. Prediction based solutions are, to the best of our knowledge, the only possible way to compensate for the dead-time [26]. The Smith predictor and MPC are two solutions in this domain which have the advantage of providing a large prediction horizon. Nevertheless, they both suffer from sensitivity to plant modelling errors [26].

2.3 Software Used for Design and Simulation

Simulation based experiments elaborated in Chapter 7 employ MATLAB, Simulink, and TrueTime software tools. The relevant features of these tools are highlighted in this Section.

2.3.1 MATLAB and Simulink

The MATLAB and Simulink software applications are developed by Mathworks [27]. MATLAB's main purpose is to perform numerical computations required not only in the scientific domain but also in the financial domain. MATLAB incorporates a high-level language to develop scripts which are executed in MATLAB's interpreter. Simulink is packaged with MATLAB and is invoked through MATLAB or used as a standalone application. MATLAB along with Simulink can be used to design and simulate digital control systems.

Introduced in its 2011 release, MATLAB and Simulink added a feature to automatically tune a PID controller which can either be in continuous-time or discrete-time. The tuning algorithm used by MATLAB and Simulink is not available in the public domain. In MATLAB the PID controller is tuned via the command line interface [28], while in Simulink the PID controller is tuned using a graphical interface [29]. The graphical interface is invoked through the PID controller simulation block. The slider in the graphical interface is used to tune the output response of the plant connected to the PID controller. A screenshot of the graphical interface is shown in Figure 2.3.

Performance metrics including rise time, settling time, and percent overshoot change in response to changing the position of the slider. By observing performance metrics reported by the graphical interface, a control system designer can design a PID controller that meets performance requirements. This research utilizes Simulink's graphical interface to tune PID controllers.

Simulink's discrete-time PID controller block implements the transfer function given by 2.7 [30], and this form is used in this research.

$$C(z) = P + I \frac{h}{z-1} + D \frac{N}{1 + N_D h \frac{1}{z-1}} \quad (2.7)$$

The P , I , and D terms are constants, h is the sampling period, and N_D is the derivative filter coefficient. The graphical tuner updates the P , I , D , and N_D terms

Figure 2.3: Simulink PID Controller Tuner Interface

when the slider is moved. Both the integral and derivative terms use the forward Euler approximation method. This realization is consistent with implementations in the literature [3] [31].

2.3.2 TrueTime

TrueTime is a framework that leverages Simulink's simulation platform, and it simulates concurrently executing control tasks using a real-time kernel [32]. TrueTime exists as a separate Simulink simulation block that can be simulated along with Simulink's native simulation blocks. This simulation block is defined as the *TrueTime kernel* in [32]. The real-time kernel in TrueTime implements the EDF and RM scheduling algorithms and gives the option to schedule tasks in a preemptive or non-preemptive manner. Using TrueTime as a base, periodically executing control tasks can be developed to execute actual control algorithms. Apart from periodic tasks, TrueTime provides the capability to execute interrupt service routines (ISRs). In its current release TrueTime only allows servicing of timer interrupts. When developing

tasks including ISRs, the developer has to specify the execution time. If a non-zero value is specified for the RTOS context switching and scheduler delay, TrueTime simulates the context switching and scheduler overhead of the kernel. TrueTime uses Simulink's notion of runtime to simulate the passage of time for the tasks. As control tasks, ISRs, and RTOS execute in parallel, they each interfere with each other's execution causing a non-determinism in the execution of the control tasks. This is the manner in which jitter is simulated in TrueTime.

Aside from developing and simulating control tasks, TrueTime provides the ability to log statistics on tasks scheduled by the scheduler. These statistics include release time, response time, start latency, and execution time [33]. Furthermore, the TrueTime kernel simulation block has a Scheduler output port, which can be connected to a Simulink scope block. This allows the execution history (releases, execution, and termination) to be graphically viewed for control tasks, ISRs, and RTOS.

2.4 Example Plants Used for Performance Evaluation

This Section describes three example plants found in the literature to build digital control systems and evaluate the performance of the solution presented in the thesis. These plants are: hard disk servo, voltage stabilizer, and a plant that represents an industrial process. The design of the discrete-time PID controller to control these plants are described in Chapter 6, and the performance requirements of each digital control system is presented in Chapter 7, Section 7.1.

2.4.1 Hard Disk Servo

A hard disk drive has disks coated with a thin magnetic layer also known as the recording medium [34], [35]. The data on the disks are organized into concentric

circles which are referred to as tracks. The disks are rotated at a high speed. The data on the disk is written or read using a read/write head. This read/write head is attached to an arm which is actuated by a voice coil motor (VCM), and it moves in the radial direction from one track to the other.

The plant of a hard disk drive consists of the VCM and the attached arm with the read/write head. A high-level diagram of the plant re-drawn from [36] is shown in Figure 2.4.

Figure 2.4: Hard Disk Servo Plant

The differential equations that govern the electromechanical dynamics of the VCM are given by (2.8) [35].

$$\begin{aligned} J \frac{d^2\theta}{dt^2} + C \frac{d\theta}{dt} &= K_i i \\ L \frac{di}{dt} + Ri &= V - K_i \frac{d\theta}{dt} \end{aligned} \quad (2.8)$$

J is the inertia of the arm and read/write head, C is the viscous damping coefficient of the bearings, K_i is the electromotive force constant of the VCM, θ is the angular position of the arm, L is the electrical inductance, R is the electrical resistance, i is the current to the VCM, and V is the voltage to the VCM.

Applying the Laplace transform to (2.8), the transfer function for the input voltage to output angular position is given in (2.9) [35].

$$G(s) = \frac{\theta(s)}{V(s)} = \frac{K_i}{s(Js + C)(Ls + R) + K_i^2 s} \quad (2.9)$$

The plant model can be made more accurate by including high resonance modes [34], [37]. However, this makes the plant model non-linear and thereby hard to control. Therefore, the model given in (2.9) is sufficient for the scope of this research.

The following values were substituted to (2.9) to obtain the transfer function that will be used to evaluate the performance: $J = 8.0 \times 10^{-13} \text{ kgm}^2$, $C = 2.5 \times 10^{-4} \text{ Nm/rads}^{-1}$, $K_i = 0.02 \text{ Nm/rad}$, $L = 1.0 \times 10^{-5} \text{ H}$, $R = 0.5 \Omega$ [36]. This transfer function is given by (2.10).

$$G(s) = \frac{\theta(s)}{V(s)} = \frac{0.02}{8.0 \times 10^{-13} s^3 + 4.25 \times 10^{-8} s^2 + 5.25 \times 10^{-4} s} \quad (2.10)$$

2.4.2 Voltage Stabilizer

The voltage stabilizer used for the performance evaluation is in the form of a RCRC circuit, the model of the plant was obtained from [38] and [39]. The high-level diagram of the plant re-drawn from [38] is shown in Figure 2.5.

Figure 2.5: Voltage Stabilizer Plant

The differential equations that model the plant given are by (2.11),

$$\begin{aligned}
\dot{q}_1 R + (q_1 - q_2) \frac{1}{C} &= V_{in} \\
\dot{q}_2 R + (q_2 - q_1) \frac{1}{C} + q_2 \frac{1}{C} &= 0 \\
q_2 \frac{1}{C} &= V_{out}
\end{aligned} \tag{2.11}$$

The state-space equations that model the plant are given by (2.12). The constant values in the equation were derived by substituting the following values: $R = 330K\Omega$ and $C = 100nF$ to the state-space equations.

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -918.27 & -90.90 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 918.27 \end{bmatrix} u(t) \tag{2.12}$$

$$y(t) = [1 \ 0] x(t)$$

The transfer function of the plant given by (2.13) was obtained by executing the `ss2tf` [40] command in MATLAB.

$$G(s) = \frac{918.27}{s^2 + 90.9s + 918.27} \tag{2.13}$$

2.4.3 An Industrial Process

Industrial processes typically find their application in the manufacturing industry [41]. Control systems are employed in industrial processes to automatically control process variables such as heat, pressure, temperature, etc. Examples of industrial processes include hydraulic systems, chemical plants, steam turbine systems, etc. The transfer function that models a representative industrial process is given by (2.14) and its expansion is given by (2.15). This model does not correspond to a specific industrial process and it was obtained from [42]. The original model contains a dead-time of 400ms which is not used in this research.

$$G(s) = \frac{1}{(s+1)(0.2s+1)(0.04s+1)(0.008s+1)} \quad (2.14)$$

$$G(s) = \frac{1}{6.41 \times 10^{-5}s^4 + 9.984 \times 10^{-3}s^3 + 0.2579s^2 + 1.248s + 1} \quad (2.15)$$

Chapter 3

State-of-the-Art in Improving Performance of Digital Control Systems Due to Software Induced Delays

The literature encompasses three broad categories of solutions to improve the performance of digital control systems due to software induced delays, and they are, compensating the delays from a control systems perspective, reducing delays from a real-time systems perspective, a hybrid approach that uses both control systems and real-time perspectives. Solutions belonging to these three categories are described in this Chapter.

Furthermore, the categories of the ways in which performance is assessed in the literature are fourfold:

1. Estimating performance using a cost function and comparison between a reference and/or other solutions in the literature: this type of assessment is defined as *PerfCost*.

2. Qualitative assessment of the plant response graphically without quantifying performance: this type of assessment is defined as *PerfQual*.
3. Measuring performance using metrics that characterize the plant response; performance requirements are defined and based upon these metrics; a cost function can also be used to estimate performance: this type of assessment is defined as *PerfPlantReqs*.
4. Measuring performance using software related performance metrics such as jitter and input to output delay, occasionally, a cost function is also used: this type of assessment is defined as *PerfSW*.

3.1 Compensating Delays in the Digital Controller

The type of delays compensated by the solutions presented in the literature include software induced jitter and constant delays.

3.1.1 Compensating for a Constant Delay

Åström et al. provides a solution that is applicable in the design of any kind of digital controller [1]. The author suggests to incorporate any constant time delay between input sampling and the response of the plant in the process (plant) model, when designing the controller. The solution is equally applicable in the design of plant dead-time compensators [43]. The solution considers a continuous time control system described by (3.1),

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t - \kappa) \quad (3.1)$$

Initially, the solution assumes $0 \leq \kappa \leq h$ where h is the sampling period and κ is the constant time delay. Subsequently, the author extends the solution to cover the

case where $\kappa > h$. The zero-order hold discrete-time state estimation of the plant with the delay is given by (3.2),

$$x(kh + h) = e^{Ah}x(kh) + \int_0^{h-\kappa} e^{As} ds Bu(kh) + e^{A(h-\kappa)} \int_0^{\kappa} e^{As} ds Bu(kh - h) \quad (3.2)$$

Two approaches are recommended by the author to design a solution to handle a delay: postpone outputting the control law until the start of the next period ($\kappa = h$) and output the control law immediately after it is ready ($\kappa < h$). Furthermore, the author advises to include the delay (h in the first and $\kappa < h$ in the second recommendation) in the process model as accomplished in (3.1), (3.2). Equation (3.1) implies that the solution is meant to handle a constant delay. Therefore, it is not clear how the author intends to include a time varying delay in the process model for the second recommendation.

The Thiran all pass filter is used in the digital signal processing domain to model a constant delay that is a fraction of the sampling period [44]. From a survey of the literature this method has not been applied for digital control systems. However, Mathworks (creators of MATLAB and Simulink) has implemented a Thiran filter to model constant delays between input sampling to outputting the control signal, that are a fraction of the sampling period for digital control systems [45].

A Mathworks tutorial described in [46] demonstrates how to design a PID controller to compensate for a constant input to output delay. The design uses the solution proposed by Åström et al. where the constant input to output delay is included in the process model, and the delay is equal to one sampling period. The modeled delay is considered to be the worst case delay. The tutorial does not demonstrate the efficacy of the solution in the presence of jitter.

Gambier proposed a simple prediction based solution to compensate for a constant delay between input sampling to outputting the control signal in [2]. The simple

prediction solution uses linear extrapolation to predict the control law in the future. Control law calculation is performed in the previous period and output at the start of the next period. In spite of the simplicity of the solution, the author suggests the prediction can be improved by incorporating a model of the plant.

Bibian et al. [47] proposed two prediction based solutions viz. modified predictor and simplified predictor to compensate for a constant delay between input sampling and outputting of the control signal by using linear extrapolation for the prediction. The authors developed the solution in the context of direct current switchmode power supply digital control systems that compensate for direct current bus ripple. Performance evaluation results reveal that the plant response of both solutions satisfy the percent overshoot performance requirement. When comparing the two solutions, the authors mention that the simplified predictor is computationally less demanding than the modified predictor. According to the authors the disadvantage of the modified predictor is the prediction is sensitive to plant model errors. The control performance of the simplified predictor is inferior to the modified predictor and it does not require a model of the plant.

PerfCost is the method used by the majority of the solutions in this Section to assess performance. On the other hand, to the best of our knowledge, PerfPlantReq is only used by Bibian et al. (please see [47]) in the entire literature of improving performance of digital control systems in the presence of software induced delays.

3.1.2 Compensating for Jitter

The work by Yu et al. [42] investigates the effects of sampling jitter on the performance of digital control systems for industrial processes. Sampling jitter is created in their experimental model using a stochastic process. The authors demonstrate that the performance of PID controllers improve when a low pass filter is used in conjunction with the derivative component.

The research by Kao et al. proposes a criterion to determine if a SISO digital

control system with input-output jitter can be stable or not [48]. This research has been extended by Cervin in [49] to take into account the combined effects of sampling jitter, output jitter, and constant computational delay. The authors use a loop transformation technique to model jitter for a digital control system with a continuous-time plant. The small gain theorem [50] is used as a basis of the authors' solution and it can be used as a stability check for control systems.

Skaf et al. in [51] has proposed a solution based on linear matrix inequalities to formulate the upper and lower bounds for performance degradation for a digital control system in the presence of sampling jitter. Additionally, the author uses the linear matrix inequalities to derive a constant state feedback controller that reduces the performance degradation bounds.

JitterBug is a MATLAB based tool developed by Cervin et al., which can be used to ascertain the sensitivity of a digital control system to various types of jitters and constant computational delay [52] without resorting to simulation. In addition, the tool can be used to determine the effectiveness of jitter compensating controllers. The tool expects a user defined cost function to evaluate the performance of the digital control system. Another use of the tool is to synthesize digital LQG controllers.

Smeds et al. have developed analytical models to predict the effects of sampling jitter and input-output jitter on positioning error for motion digital control systems [53]. In addition, they have also proposed a solution to compensate for input-output jitter by cascading an add-on compensator to the main controller. The cascaded compensator attenuates the controller gain near the system Nyquist frequency $\omega_N = \pi/h$, where h is the sampling period. This is accomplished by adding a zero at ω_N .

Marti et al. have proposed a solution to compensate for sampling jitter in PID and state feedback controllers [18]. The authors suggest to first measure the sampling interval, and then update the control parameters using the sampling interval as an input parameter. Updating the control parameters can be performed online or offline [54] [55]. The disadvantage of the online method is an additional and possibly

variable computational overhead. In the offline method all possible controller parameter updates are calculated offline and stored in a look-up table. The downside of this approach is it can occupy a large amount of data memory, which is an issue for computer systems with a limited amount of memory.

Marti et al. proposed another solution to compensate for both sampling jitter and input-output jitter in [56]. This solution relies on a model of the plant to predict the control signal and it is intended for state feedback controllers. To apply the solution the authors measure the sampling interval and the input to output delay in the same period. However, the method used to measure the input to output delay is not clear from the paper. This is because it is impossible to determine how much time is required for the control law computation just after sampling the inputs. The disadvantages of this solution are inaccuracies in the prediction manifested through errors in modeling the plant and computationally intensive operations to perform the prediction.

The research performed by Niculae et al. [57] propose a solution to compensate for sampling jitter. However, the proposed solution is very much similar to the solution proposed by Marti et al. in [18].

The solution proposed by Lincoln in [58] is aimed at compensating input-output jitter. Initially, the *delay error* which is defined in the paper as the difference between the actual control signal and the desired control signal is measured. The delay error is passed through two continuous-time systems. The disadvantage of this solution is its impracticality, that is, the solution requires designing additional continuous-time components.

The research by Nilsson et al. [59] is based on digital network control systems. The authors point out that the solution can also be applied to single node digital control systems, and it is primarily applicable in the compensation of randomly varying input-output jitter. Optimal stochastic control is the principle behind the solution and it assumes a probability distribution of the jitters. The disadvantage of this solution is

it requires a priori knowledge of the jitters in the system.

The one shot task model solution by Lozoya et al. [38] compensates for input-output jitter using prediction. According to the solution, the control law calculations are predicted in the current period after sampling the inputs, and output precisely at the start of the next period. An observer based state feedback control strategy is used to control the plant. Therefore, the prediction relies on a model of the plant to calculate the required states. The authors do not have strict requirements on input sampling and allow for sampling jitter. The time difference between sampling the inputs and the start of the next period is used to predict the control law applied in the future. This solution also shares the same disadvantages in the jitter compensation solution presented by Marti et al. in [56]. That is, computationally intensive operations are necessary for the prediction and the possibility of errors in the prediction due to inaccuracies modelling of the plant. The performance of the solution was evaluated using a real implementation of a voltage stabilizer digital control system.

Ovaska et al. have proposed a prediction based solution to compensate for input-output jitter [21]. In their solution, input sampling and the predicted control law calculated from the previous period, is output to the plant simultaneously at the start of the period. The rest of the period is utilized to calculate the control law for the next period. The prediction is performed using a linear infinite impulse response polynomial.

As with Section 3.1.1, most solutions in this Section use PerfCost to assess performance. Solutions described in [18], [54], [55], [56] use PerfQual for the assessment.

3.2 Reducing Delays

Solutions that involve reducing software induced delays can be broadly categorized as software and hardware based solutions. In software based solutions, modifications are made to the task model and scheduling algorithms with the aim of reducing jitter

due to scheduling multiple tasks on the same processor. Hardware based solutions are proposed to synchronize input sampling and outputting the control law to a specific timing boundary.

3.2.1 Software Based Solutions

Buttazzo et al. [17] compares three different solutions used for jitter minimization. Non-preemptive scheduling, deadline reduction, and bounding inputting and outputting to the start and end of the sampling period respectively via task splitting are the three solutions. The authors do not suggest which solution provides the best jitter minimization, however, the deadline reduction solution had the best overall performance estimated via a cost function. In addition, the task splitting solution showed a constant performance degradation at the expense of a large input to output delay.

Solutions described in this Subsection all use PerfSW method to assess performance. A cost function is not used in all solutions in this Section to quantify performance. The remainder of this Subsection categorizes and discusses software based solutions in the literature from the real-time systems perspective to reduce delays.

3.2.1.1 Relative Deadline Assignment

According to J.Liu, if the relative deadline of a periodic task is reduced to be less than its period, the end result is a reduction in the response time jitter (in the context of this research response time jitter is equivalent to output jitter) [16]. The idea is to reduce the slack time, which is the duration between the task response time and relative deadline. A reduction in the slack time means that the task now has less room to vary its response, which leads to a reduction in response time jitter. In spite of reducing response jitter, the downside of this approach is the schedulability of the tasks can be impacted. On the other hand, this solution requires a low implementation

cost compared to other approaches.

The majority of relative deadline reduction algorithms found in the literature are calculated offline [60], [61], [62], [19], while the solution in [63] uses an online algorithm. A hybrid approach is adopted in [64] where the deadlines are initially calculated offline and changed online as the workload increases. The work described in [65] is different compared to other solutions in the sense that, response time jitter requirements are used to derive the relative deadlines.

3.2.1.2 Task Splitting

The principle behind task splitting is to partition the periodic control task into two or more subtasks with the aim of reducing software induced jitter. Furthermore, task parameters such as priority, relative deadline, and offset are varied among subtasks. Typically, sampling and output functions are allocated to high priority subtasks with very short execution times, while control law calculation is assigned to lower priority subtasks.

The research in [66], [67], [68] uses a solution called initial mandatory final (IMF) to split tasks, while [52] uses Control Output Update State (COUS). A hybrid approach, which is a combination of both the IMF and COUS, is described in [69].

The advantage of this approach is the best overall minimization of software induced jitter compared to other solutions. However, an undesirable outcome is an increase in the input to output delay due to bounding the input and output to fixed instants in time. This situation as illustrated by Åström et al. in [1] and described in Section 3.1.1 can be a bad design choice for digital control systems.

3.2.1.3 Changing the Scheduling Algorithm

The aim of this class of solutions is to modify the underlying scheduling algorithm in such a way that it leads to a reduction in software induced jitter. The simplest solution is described in [70] where a non-preemptive scheduler is used. Conversely,

solutions presented in [71], [72], modify the task schedule generated by scheduling policies such as EDF and DM.

Compared to other approaches, the solution described in [73] proposes a new scheduling algorithm called distance-constrained task system (DCTS). The idea behind DCTS is to place a timing constraint between the completion of successive jobs of a task [74]. A different approach is described in [75] where the preemption segment of a task is changed statically to minimize software induced jitter.

3.2.1.4 Dynamic Voltage Scaling

Dynamic voltage scaling (DVS) is another solution which can be used to minimize jitter. The purpose of DVS is to find a trade-off between performance and power consumption by changing the voltage of the processor dynamically [76]. As the voltage is scaled, the operating frequency of the processor is also changed. As a result, there is a change in the execution time of the software executed by the processor. DVS when used incorrectly can also increase jitter [77].

The DVS based solution by Mochocki et al. [78] is intended to minimize completion time jitter (also known as response time jitter) for tasks scheduled using fixed priority scheduling.

3.2.1.5 Comparison of Solutions Types

This Subsection summarizes and compares the four software based solution types to reduce delays and jitter described in Section 3.2.1, and is presented in Table 3.1. The abbreviations used in Table 3.1 are summarized in the following list:

- **RDA**: Relative Deadline Assignment
- **TS**: Task Splitting
- **CSA**: Changing Scheduling Algorithm

Solution	Main Feature	Advantages	Disadvantages
RDA	Reducing the deadline leads to a reduction in slack time which results in jitter minimization	Low implementation cost	The schedulability of software tasks is reduced
TS	The workflow of a digital control system is split into 2 or more subtasks	Best overall software jitter minimization	1. The amount of context switches increase, which increases the total overhead 2. Long artificial delays between input and output
CSA	Changes are made to the scheduling algorithm to minimize jitter	Best overall input to output delay reduction	Implementation complexity
DVS	The execution time of tasks are changed dynamically to minimize jitter	Simultaneous reduction of both power consumption and jitter	1. Not all processors support DVS 2. If not applied properly jitter increases

Table 3.1: Comparison of Software Based Solutions That Reduce Delays and Jitter

3.2.2 Hardware Based Solutions

The amount of hardware based solutions in the literature in comparison to software based solutions are less. The work by Richter et al. [79] delays the notification that the inputs have been sampled or output has been calculated, by using hardware buffers which is the same as elimination of jitter. However, as pointed out by Åström et al. in [1] and highlighted in Section 3.1.1, the introduction of a long artificial delay from input sampling to outputting the control signal is not a recommended approach for reducing jitter in digital control systems.

Halang [80] presented a method that makes use of timed peripherals to eliminate jitter. These timed peripherals can be programmed to be activated after a certain delay. As a result, jitter is eliminated. However, the author has not backed up the claims made in the publication with experimental data. This solution also shares the same problem as with Richter et al. in [79] where a long artificial delay is used to eliminate jitter.

Contemporary microcontrollers provide the capability to trigger an ADC and/or a DAC without software intervention. Almost all microcontrollers have a continuous operating mode for the ADC. In contrast, only a few microcontrollers include the option to trigger the DAC automatically. Microcontrollers by Atmel (AT91SAM) [81] and Standard Microelectronics (STM32F405xx and STM32F407xx series) [82] have a DAC that can be triggered by an external trigger.

3.3 Compensating for Delays Combined with Delay Reduction

Integrating approaches from the real-time systems perspective and control systems perspective is known as control and scheduling co-design [22]. Similar to software and hardware co-design, real-time systems approaches and control systems approaches are

applied in a synergistic manner to produce multiple digital controllers executing as real-time tasks simultaneously on a computer with limited resources to achieve the best possible performance. The majority of solutions in the literature utilized the PerfCost method to assess performance, while the solution described in [83] made use of PerfSW.

Cervin et al. in [84] formulated the concept of jitter margin, which takes the constant computational delay of a control task as the input parameter and outputs the maximum allowable input-output jitter to keep the digital control system stable. The stability margin developed by Kao et al. in [48] is used as a basis in the authors' formulation. The authors propose a solution that leverages jitter margin to assign appropriate relative deadlines to control tasks. Additionally, an iterative control and scheduling co-design procedure that utilizes jitter margin is proposed by the authors to derive sampling periods for digital controllers so that each controller degrades uniformly due to input-output jitter.

In a different research, Cervin et al. propose the control server approach as a control and scheduling co-design solution for both digital control systems and signal processing applications [85]. The control server approach uses processor utilization to link the real-time systems and control systems aspects of the design. EDF is used in the solution to schedule tasks. Cervin's COUS model in [52] is used to split a control task into two subtasks. Furthermore, application specific cost functions that takes the processor utilization of the control task as an input, are used to calculate the performance of the designed controller.

Behnam et al. in [83] have proposed an iterative control and scheduling co-design solution to design digital controllers to maximize performance and minimize input-output jitter. The authors initially use the jitter margin proposed by Cervin et al. in [84] to determine the maximum allowable input-output jitter, and as a result, suitable deadlines for the control tasks. Optimal performance of the digital controllers is achieved by optimizing the apparent phase margin. The target of the solution is

to find the optimal sampling periods for a set of LQG continuous-time controllers. Finally, the iterative solution checks if the control tasks with their current sampling periods are schedulable and the scheduling of the tasks are determined a priori.

The research by Wu et al. in [86] propose a control and scheduling co-design solution to design digital controllers by selecting optimal sampling periods and relative deadlines. The sampling period and relative deadlines are derived through optimization by minimizing a cost function where both parameters are taken into account.

Aminifar et al. have proposed a control and scheduling co-design solution for digital network control systems, which can also be applied to a single node digital control system [87]. Initially, the solution uses optimization to derive the optimal sampling periods and priorities for the control tasks. Subsequently, LQG controllers are designed to compensate for the known constant computational delay. The performance of the synthesized controllers are evaluated using a cost function. The solution is applied iteratively until optimal performance is achieved. The authors point out that using the worst case scenario for a design procedure yields sub-optimal control quality as the worst case only occurs rarely.

Chapter 4

A Novel Solution to Improve Performance of Digital Control Systems Due to Software Induced Jitter

Based on the shortcomings found in the state-of-the-art reviewed in Chapter 3, this Chapter provides the motivation for the research carried out to develop a new solution in order to solve the issue of performance degradation in digital control systems due to jitter. This Chapter initially analyzes the state of the art and discusses their shortcomings, the thesis claim is subsequently presented which contains the problem solved by this research and the proposed solution, along with the boundaries of the research. A summary of the steps involved in proving the thesis claim, and the contributions to knowledge are also presented.

4.1 Analysis of the State-of-the-Art

Chapter 2, Section 2.2 highlights the effects of software induced delays on a digital control system, which essentially is a performance degradation. A detailed survey of the state-of-the-art solutions that improve performance in digital control systems due to software induced delays is provided in Chapter 3. The entire spectrum of solutions were proposed from three different perspectives. Firstly, the control systems perspective attempts to solve the issue of performance degradation by compensating for delays in the digital controller. The second perspective is from the real-time systems perspective which attempts to solve the issue of performance degradation by improving the scheduling of the control tasks with the aim of reducing delays as much as possible so that it would eventually lead to an improve in performance. Finally, the third perspective is a combination of both perspectives that considers scheduling of control tasks and ways to improve the digital controller to obtain the best possible performance in the presence of delays. The third perspective is also referred to as control and scheduling co-design.

Criteria used to analyze the literature in Chapter 3 are discussed in detail in the following list:

Criteria 1: A solution that uses a PID controller for the compensation

- Despite the wide spread use of PID controllers in the industry, the majority of solutions in the literature are not intended for PID controllers. The majority of solutions target LQG controllers and state feedback controllers with observers. The only solution, to the best of our knowledge, found in the literature intended for PID controllers was presented by Marti et al. in [18]. This solution only compensates for sampling jitter by computing the P , I , and D parameters after sampling the inputs in each period. The main disadvantage of this approach is the computational overhead to recalculate the control parameters. To improve the run-time overhead the authors suggested to use a lookup table containing pre-calculated control parameters.

However, this approach requires a large memory space to store the table. The authors predict that the size of the table could be in the order of several KBs. Usage of large look up tables becomes an issue only for computer systems with limited memory resources.

Modelling a constant delay in a digital control system was proposed by Åström et al. [1] and described in Chapter 3, Section 3.1.1. The authors point out that a constant delay compensating controller can be designed by using this approach. Several solutions in the literature have adopted this approach to compensate only for the constant computational delay portion of the overall input-output jitter and have applied it to LQG in [87] and state feedback controllers in [84]. Nevertheless, in the literature this solution has not been applied for PID controllers to compensate for software induced jitter.

Criteria 2: Satisfying performance requirements - Another shortcoming found in the literature is the fact that most solutions do not consider satisfying performance requirements based on characterizing the plant response. As discussed in Chapter 3, the majority of solutions do not employ the PerfPlantReq method to assess performance. Plant response characterizing performance requirements considered in this research are, percent overshoot, rise time, and settling time. One advantage of satisfying these performance requirements is, it always establishes the stability requirement of a digital control system. All solutions stemming from all perspectives ensure stability in a wide range of scenarios and this is important. Nevertheless, there can always be a situation where the digital control system is stable but performance requirements are not satisfied due to software induced delays. To the best of our knowledge, only Bibian et al. in [47] used the PerfPlantReqs to assess performance. The percent overshoot was only used as the performance requirement in their work.

Criteria 3: Reducing Input to Output Delay - In some publications, elimination of jitter was considered to be an important achievement [79] [80]. It is true

in some domains, such as multimedia. In the case of digital control systems elimination of jitter does not necessarily equate to the best possible performance. What is more important is reducing the sampling to output delay. For example, if a solution eliminates output jitter by introducing a large artificial constant delay (typically less than one sampling period), it does not yield good performance [1], [17]. Sampling jitter is easily eliminated in contemporary embedded microcontrollers that have ADCs with automatic triggering. Reducing sampling jitter and input-output jitter by task splitting increases the number of tasks, which also increases the amount of context switching, ultimately resulting in an increase in the aggregate context switching overhead. On the other hand, reducing the deadline for control tasks is more useful over other approaches from the real-time systems perspective [65], [17]. In addition, merely satisfying timing constraints of the real-time software for the digital control system does not guarantee that plant response characterizing performance requirements will be met. Therefore, for overall better control performance it is necessary to combine compensation of delays in the digital controller with a reduction of delays, which are accomplished in control and scheduling co-design approaches [22].

The main shortcoming of the solutions from the real-time systems perspective is a reduced emphasis on the performance of the digital controller. While it is necessary to reduce delays to achieve better performance, this is not a sufficient condition.

Criteria 4: Implementation complexity - The issue of implementation complexity is most prominent in solutions from the control systems perspective. Using a state feedback controller with a state observer used by Lozoya et al. in [38] and Cervin in [84] requires an implementation of a model of the plant in the digital controller. Usually, the amount of computations required to calculate the control signal in state feedback control with an observer is significant compared to PID controllers because of matrix computations. Another disadvantage of having to use a model of the plant in the controller is, the issue of modelling errors [26] and in turn it could

lead to inaccuracies in the calculation of the control signal. Solutions that employ LQG controllers require knowledge of optimization and that entails formulation of an appropriate cost function. Therefore, an advanced knowledge of control systems is required.

The disadvantage of PID controllers is its applicability can be limited depending on the order of the plant. The main disadvantage of LQG control over PID is the complexity in the design process. Since state feedback controllers with an observer use a model of the plant, it has the potential to provide the best possible prediction. In spite of that, the prediction could be undermined by model inaccuracies. Therefore, it can be concluded that compared to other types of digital controllers, PID controllers trade performance for reduced implementation complexity. However, the wide use of PID controllers implies the required level of performance can easily be achieved by using an easy to implement digital controller.

Criteria 5: Execution time overhead - If a solution increases the execution time overhead it could reduce the schedulability of the tasks because part of the available timing budget has to be allocated towards the execution of the solution. For example, the overhead for the state estimation in state feedback control with an observer can be significant depending on the order of the plant. Cervin in [84] and Lozoya et al. in [38] discussed this issue in conjunction with state feedback controllers. The research by Bibian et al. in [47] also indicated that their linear extrapolation solution can be computationally intensive.

In summary, a review of the state-of-the-art reveals a lack of solution to compensate for input-output jitter for PID controllers. Furthermore, a large portion of solutions do not prioritize satisfying plant response characterizing performance requirements. With this conclusion in mind, Section 4.2 presents the issue addressed by this research and a novel solution.

4.2 The Thesis

Software induced sampling jitter, input-output jitter, and output jitter can degrade the performance of a digital control system. In the worst case it can lead to failing to meet performance requirements and create instability. Current solutions have improved jitter related performance notwithstanding shortcomings which include, not considering plant response characterizing performance requirements, and an increase in implementation complexity and execution overhead. These shortcomings are presented in Section 4.1.

Statement of Thesis: *Digital control systems that utilize a PID controller can compensate for jitter by redesigning the PID controller. The compensation involves modelling a one sampling period constant delay placed in the forward loop between the plant and PID controller, and designing the P , I , and D constants for the delayed model. The result is a jitter compensating PID controller that meets performance requirements, but does not change the implementation complexity, or execution time.*

4.3 Thesis Scope

The research discussed in this dissertation is scoped to address a specific problem, the following list defines how this research is scoped:

- Software induced jitters are only considered. One exception to this is, hardware induced jitter from the ADC and DAC are also considered.
- Plant dead-time is assumed to be zero.
- The entire digital control system exists on one node. That is, the research does not consider networked digital control systems.

- The research is only concerned with the scenario where the response time $R_{i,j}$ of job $J_{i,j}$ does not exceed the sampling period h_i . This assumption is defined as the **one sampling period delay assumption** in this dissertation.
- Linear plant systems are only considered in this research and non-linear plant systems are not considered. Saturation of the plant response is not considered.
- MIMO systems are not considered while independently executing multiple SISO systems are only considered.
- External disturbances to the digital control system and execution of sporadic tasks are not considered.

In summary, this research is scoped to take into account of a system comprised of multiple independent SISO digital control systems executing in parallel. Each digital controller of a digital control system is implemented as a control task which adheres to the one sampling period delay assumption. The upper bound of the control task response is guaranteed, therefore, a comprehensive validation procedure is not required to ensure that the control tasks meet their deadlines. Furthermore, random external disturbances and transient overload conditions are not introduced to the digital control system set. Consequently, the behavior of the digital control system, from a real-time systems perspective, is predictable. The notion of predictability in the context of this research is discussed in detail in Chapter 5, Section 5.2.

4.4 Proving the Thesis Claim

In order to prove the thesis claim presented in Section 4.2, the initial step is to develop the model of the system that executes the jitter compensating PID controller. Because the proposed solution is intended for software induced delays the model also includes timing details. Chapter 5 describes this model in detail. The next step is

the design of the jitter compensating PID controller which is discussed in Chapter 6. Finally, the effectiveness of the proposed jitter compensating PID controller is evaluated through simulation experiments using example plants introduced in Chapter 2, Section 2.4. The performance in response to a step input and square wave are measured. Plant response characterizing metrics are used to measure the step input related performance, which consists of: percent overshoot, rise time, and settling time. The IAE cost function is also used to estimate the step input related performance. The tracking performance is estimated exclusively using the IAE cost function. The performance (step input and tracking) of the digital control system with the jitter compensating PID controller is compared against the same digital control system with an uncompensated PID controller. Additionally, the performance is compared against a digital control system with the proposed jitter compensating PID controller that outputs at the start of the next sampling period.

Performance results from the digital control system examples with an uncompensated PID controller will show that one or more performance requirements have been violated. On the other hand, the performance results of the digital control system examples with a compensated PID controller will show that all performance requirements have been met. Additionally, the jitter compensating PID controller will indicate better tracking performance over its uncompensated counterpart. Furthermore, performance results will also indicate better performance than the solution that outputs at the start of the next sampling period. These performance results will validate that the proposed solution can mitigate the detrimental effects of software induced jitter and ensure performance requirements can be met. Complete details of the performance evaluation and results are illustrated in Chapter 7.

Chapter 5

Modelling the System

The purpose of this Chapter is to elaborate on the model of the digital control system set that is used in this research. The term *digital control system set* as introduced in Chapter 2 refers to a collection of independently executing SISO digital control systems. All digital controllers in the digital control system set execute simultaneously on the same processor. The objective of the model is to describe the platform that executes the proposed jitter compensating PID controller and describe how delays, including jitter are created. Details of the proposed PID controller are provided in Chapter 6. In addition to describing the model, this Chapter also discusses the timing of the overall digital control system set and each digital control system. This Chapter also describes how the composition and timing of the digital control system set and each digital control system have been further refined to represent four realistic ways of implementing digital control systems. A realistic way of implementing a digital control system is referred to as a configuration in this dissertation. The performance evaluation of the proposed solution outlined in Chapter 7 utilizes these configurations. Finally, this Chapter compares these four configurations.

5.1 System Composition

The digital control system set considered in this research comprises of n SISO digital control systems implemented by n periodic control tasks: $T_i \in \{T_1, T_2, \dots, T_n\}$. The task model used in this dissertation is briefly discussed in Chapter 2, Section 2.1.5. Each control task T_i is instantiated by an infinite sequence of jobs $J_{i,j} \in \{J_{i,1}, J_{i,2}, \dots\}$. A job $J_{i,j}$ is released at release time $r_{i,j}$ by a periodic hardware timer interrupt. All n digital control systems in the digital control system set use PID controllers to control the plant. The workflow of each control task T_i is described in the following list:

1. Sample the reference signal and plant response using ADCs to produce their discrete-time equivalents.
2. Compare the reference signal and plant response to produce the discrete-time error signal.
3. Execute the PID control law to compute the discrete-time control signal.
4. Convert the discrete-time control signal to its analog equivalent using a DAC.

Figure 5.1 shows a block diagram of the model for a digital control system in the digital control system set. The Figure refines the high-level block diagram of a digital control system depicted in Chapter 2, Figure 2.1. The refinement includes the type of controller used, which is a PID controller, and the modelling of the ADCs and DAC. The proposed solution is not limited to this case, but instead, it can be extended to cover any amount of ADCs and DACs. As introduced in Chapter 2, Section 2.1.5, the response time, $R_{i,j}$ is the amount of time it takes to produce the control signal relative to the release time $r_{i,j}$ of job $J_{i,j}$.

As an example ADC1 is used to sample the reference signal and ADC2 can be used to sample the plant response. The ADC used in the system is modeled to only sample the continuous-time signal and hold it constant over the duration of one

Figure 5.1: System Model of a Digital Control System

sampling period. Similarly, the DAC is modeled as a zero-order hold device where it outputs and holds the calculated control law constant for one sampling period. The clock depicted in Figure 5.1 is generic, and for practical systems, a faster clock will be used for the controller and slower clocks will be used for the ADC and DAC. As noted in the literature review in Chapter 3, software or hardware can be used to trigger the ADCs and DAC. The discussion of this Section is independent of the trigger method. Each digital control system has exclusive use of two separate ADCs and one DAC.

5.2 Timing Model

The timing of each digital control system is governed by the clock depicted in Figure 5.1. This clock is also used by the hardware timer to generate interrupts to release each job $J_{i,j}$. As discussed in Chapter 2, Section 2.1.5 the sampling interval $\hat{h}_{i,j}$ for task T_i is not always equal to the sampling period h_i . This is due to the presence of sampling jitter. Figure 5.1 uses sampling interval instead of sampling period to indicate the presence of sampling jitter. At zero time job $J_{i,0}$ for each control task T_i is released at the same time, and thereby, all control tasks are in phase.

All software induced delays considered in this research, are generated by executing

control tasks. No additional artificial delays are injected to the digital control system set. Additionally, by executing all control tasks on the same processor, each job $J_{i,j}$ from control task T_i interferes with the execution of jobs from other control tasks and vice versa. Furthermore, the execution of context switches (CSs), RTOS scheduler, and ISRs interfere with the execution of the control tasks. The interference ultimately results in sampling jitter, input-output jitter, and output jitter. Jitter in the A/D and D/A conversions add to the input-output jitter and output jitter.

The computational delay $\tau_{i,j}$ for control task T_i varies from one sampling period to the other, and hence, there is jitter. Jitter in $\tau_{i,j}$ arises due to the variation in the execution time of the A/D conversions, the ISR triggered after completing an A/D conversion, control law calculation, D/A conversion, and RTOS scheduler. However, as more than one control task executes, the value of $InOut_{i,j}$ will be much greater than $\tau_{i,j}$. As introduced in Chapter 4, Section 4.3, this research assumes that the response time $R_{i,j}$ for each job $J_{i,j}$ does not exceed h_i , which is also known as the one sampling period delay assumption. This timing constraint is given by (5.1),

$$R_{i,j} \leq h_i \quad (5.1)$$

Chapter 4, Section 4.3 highlighted that the behavior of a digital control system in the context of this research is predictable. The predictability considered in this research is *completion time predictability*. This notion as defined by Liu in [16] states: if $f(J_{i,j})$ is the actual completion time of job $J_{i,j}$ according to the actual schedule of the job set, $f^+(J_{i,j})$ and $f^-(J_{i,j})$ are the completion times of the maximum and minimum schedules of the job set respectively, then $J_{i,j}$ is completion time predictable if $f^-(J_{i,j}) \leq f(J_{i,j}) \leq f^+(J_{i,j})$. The value of $f^-(J_{i,j})$ in this research is equal to the best case value of $\tau_{i,j}$ and $f^+(J_{i,j})$ is equal to h_i . Therefore, the relationship for completion time predictability holds for a control task T_i . Furthermore, this research, as pointed out in Chapter 4, Section 4.3 is scoped to not consider random external

disturbances to digital control systems. Therefore, the behavior of a digital control system in this research is fully reproducible.

Calculation of the PID control law for control task T_i is carried out by a software periodic task T_i^C and instantiated by an infinite sequence of jobs $J_{i,j}^C \in \{J_{i,1}^C, J_{i,2}^C, \dots\}$. Each job $J_{i,j}^C$ is spawned within job $J_{i,j}$ and scheduled by the RTOS using the EDF scheduling policy. Usually the second A/D conversion done ISR notifies the scheduler to schedule job $J_{i,j}^C$. The release time $r_{i,j}^C$ for successive jobs of T_i^C is not equal to h_i , and it lies within the range $(0, 2h_i)$. The relative deadline of job $J_{i,j}^C$ is expressed by (5.2),

$$D_{i,j}^C = D_i - r_{i,j}^C \quad (5.2)$$

This timing constraint has been placed on each job $J_{i,j}^C$ so that the one sampling period delay assumption will hold.

5.3 System Configurations

This research uses four variations of the system timing models introduced in Sections 5.1 and 5.2, and these are referred to as **configurations** in this dissertation. A variation of the system model is due to the method used to trigger the ADCs and DAC, which are software or hardware. Therefore, this results in four configurations in total. These configurations represent realistic ways of implementing a digital control system. The proposed jitter compensating PID controller discussed in Chapter 6 is applied to each configuration and evaluated in Chapter 7. As indicated in Section 5.2, the timing of all four configurations is governed by a periodic hardware timer. Subtask T_i^C is used by all four configurations to calculate the control law. The following subsections discuss the four configurations.

5.3.1 Configuration 1: Hardware Input and Hardware Output Configuration

This configuration is also referred to as *configuration 1* in this dissertation, and it represents the most conservative approach employed to implement a digital control system. The aim of configuration 1 is to implement a digital control system, which can minimize software induced jitter as much as possible. The ADCs and DAC are triggered continuously by hardware without any software intervention. However, software may be required initially to set-up the ADCs and DAC to be triggered in a continuous manner. Both the ADCs and DAC are triggered using a periodic hardware timer. The ADCs are triggered using the hardware timer that releases a job $J_{i,j}$, while another hardware timer is used to govern the timing of triggering the DAC. In this configuration both the ADCs are synchronized to start sampling exactly at the same instant in time. The second hardware timer is set to expire at a fixed offset from release time $r_{i,j}$. The relative offset O_i for control task T_i is given by (5.3),

$$O_i = \max \{r_i^C + R_i^C\} \quad (5.3)$$

R_i^C is a vector of response times and r_i^C is a vector of release times of task T_i^C . In other words, the relative offset O_i is determined by the worst case response of T_i^C . Therefore, an initial analysis of the digital control system set is required to formulate O_i .

The timing of a job $J_{i,j}$ for control task T_i for configuration 1 is depicted in Figure 5.2 and its timing parameters are summarized in Table 5.1.

Figure 5.2: Timing Diagram of Configuration 1

Timing Parameter	Description
$\tau_{i,j}$	A/D conversion delay + $2 \times$ A/D done ISR delay + $2 \times$ RTOS CS delay + control law calculation delay + D/A conversion delay
Sampling jitter	Sampling jitter is eliminated and the sampling interval $\hat{h}_{i,j}$ for job $J_{i,j}$ always constant and equal to h_i
Input-output jitter	Input output jitter is minimized as much as possible. Jitter in the D/A conversion causes the input output jitter.
Output jitter	Output jitter is minimized as much as possible. Jitter in the D/A conversion causes the output jitter.
$R_{i,j}$	The response time is equal to O_i + D/A conversion delay

Table 5.1: Summary of Configuration 1 Timing Parameters

5.3.2 Configuration 2: Hardware Input and Software Output Configuration

In this configuration, the ADCs are triggered exactly in the same manner as in configuration 1 and, therefore, the ADCs are synchronized. In contrast to configuration 1, software is used to trigger the DAC. This configuration is referred to as *configuration 2* in this dissertation. Configuration 2 represents the most common implementation strategy for a digital control system. This is because contemporary ADCs are equipped with a continuous trigger mode, which does not require software intervention. After the control law calculation job $J_{i,j}^C$ has finished executing the control law, the last statement in the job is to trigger the DAC.

Both configuration 1 and configuration 2 have input-output jitter and output jitter. In spite of that, configuration 2 will have more of those jitter compared to configuration 1 because the DAC is triggered by software. Interference by other control tasks, ISRs, and RTOS scheduler all contribute to the increase in jitter and delays to trigger the DAC. It is reasonable to argue that the amount of jitter due to the D/A conversion is less than the jitter to trigger the DAC by software.

Figure 5.3 shows the timing of a job $J_{i,j}$ for control task T_i for configuration 2 and Table 5.2 summarizes its timing parameters.

Figure 5.3: Timing Diagram of Configuration 2

Timing Parameter	Description
$\tau_{i,j}$	A/D conversion delay + $2 \times$ A/D done ISR delay + $2 \times$ RTOS CS delay + control law calculation delay + delay to trigger DAC + D/A conversion delay
Sampling jitter	Sampling jitter is eliminated and the sampling interval $\hat{h}_{i,j}$ for job $J_{i,j}$ is always constant and equal to h_i
Input-output jitter	Input output jitter is present and $InOut_{i,j}$ is always $< h_i$ for job $J_{i,j}$
Output jitter	Output jitter is present and $Out_{i,j}$ exists within the range $(0, 2h_i)$ for job $J_{i,j}$
$R_{i,j}$	The response time can vary for job $J_{i,j}$ and is always $< h_i$

Table 5.2: Summary of Configuration 2 Timing Parameters

5.3.3 Configuration 3: Software Input and Hardware Output Configuration

ADCs are triggered by software in this configuration, which is known as *configuration 3* in this dissertation, while the DAC is triggered when a hardware timer expires. The amount of output jitter in configuration 3 is equal to configuration 1, because of jitter in the D/A conversion. On the other hand, the amount of input-output jitter is much larger compared to configuration 1, due to the presence of sampling jitter. A second hardware timer is used in this configuration to trigger the DAC similar to configuration 1. The relative offset O_i used to trigger the second hardware timer is derived exactly in the same manner as in configuration 1, and is given by (5.3). Nevertheless, O_i for configuration 3 can be longer compared to configuration 1, due to the presence of sampling jitter. As the sampling instant for job $J_{i,j}$ moves away from its release time $r_{i,j}$, the response time $R_{i,j}$ also increases. Configuration 3 is an unlikely implementation strategy because most contemporary DACs do not feature a hardware trigger feature, and since contemporary ADCs include a hardware trigger feature it makes sense to leverage it to eliminate sampling jitter.

A dedicated software periodic input task T_i^I is used to trigger the ADCs in contrast to configurations 1 and 2. Furthermore, the execution of software to trigger each individual ADC will always result in a gap between the first and second triggers. Therefore, both ADCs are not synchronized in this configuration.

T_i^I also has an infinite sequence of jobs $J_{i,j}^I \in \{J_{i,1}^I, J_{i,2}^I, \dots\}$. Similar to $J_{i,j}^C$, $J_{i,j}^I$ is also spawned within job $J_{i,j}$. The interval between the release of successive jobs for $T_{i,j}^I$ is not equal to h_i , and it lies within the range $(0, 2h_i)$. The relative deadline D_i^I of T_i^I is initially set to h_i . Subsequently, by simulating the digital control system set, an appropriate relative deadline is assigned. The relative deadline assignment for T_i^I is further discussed in Chapter 7, Section 7.2. In configuration 3, the timer ISR executes after job $J_{i,j}$ is released at $r_{i,j}$. Its sole purpose is to notify the RTOS to

schedule input job $J_{i,j}^I$. After the timer ISR completes its execution, a CS will occur and the RTOS schedules $J_{i,j}^I$ using EDF. The introduction of the input task T_i^I also doubles the amount of CSs compared to configurations 1 and 2, and consequently, the overheads increase. Using a dedicated input task to trigger the ADCs instead of performing the same functionality in the timer ISR introduces more sampling jitter to the digital control system set.

The timing of job $J_{i,j}$ for control task T_i for configuration 3 is depicted in Figure 5.4, and the timing parameters for this configuration are summarized in Table 5.3.

Figure 5.4: Timing Diagram of Configuration 3

Timing Parameter	Description
$\tau_{i,j}$	A/D conversion delay + $2 \times$ A/D done ISR delay + separation between ADC triggers + $4 \times$ RTOS CS delay + control law calculation delay + D/A conversion delay
Sampling jitter	Sampling jitter is present and the sampling interval $\hat{h}_{i,j}$ exists within the range $(0, 2h_i)$ for job $J_{i,j}$
Input-output jitter	Input output jitter is present and $InOut_{i,j}$ is always $< h_i$ for job $J_{i,j}$
Output jitter	Output jitter is minimized as much as possible. Jitter in the D/A conversion causes the output jitter.
$R_{i,j}$	The response time is always constant for job $J_{i,j}$ and is equal to O_i + D/A conversion delay

Table 5.3: Summary of Configuration 3 Timing Parameters

5.3.4 Configuration 4: Software Input and Software Output Configuration

The final configuration discussed in this dissertation, also known as *configuration 4*, shares traits from configuration 2 and configuration 3, where the ADCs and DAC are triggered exclusively by software. This configuration represents the least conservative way to implement a digital control system because jitter minimization is not emphasized. Similar to configuration 3, configuration 4 also uses a dedicated software periodic input task T_i^I to trigger the ADCs. The ADCs are not triggered in a synchronized manner.

The timing of a job $J_{i,j}$ for control task T_i for configuration 4 is illustrated in Figure 5.5 while the timing parameters for this configuration are summarized in Table 5.4.

Figure 5.5: Timing Diagram of Configuration 4

Timing Parameter	Description
$\tau_{i,j}$	A/D conversion delay + $2 \times$ A/D done ISR delay + separation between ADC triggers + $4 \times$ RTOS CS delay + control law calculation delay + delay to trigger DAC + D/A conversion delay
Sampling jitter	Sampling jitter is present and the sampling interval $\hat{h}_{i,j}$ exists within the range $(0, 2h_i)$ for job $J_{i,j}$
Input-output jitter	Input output jitter is present and $InOut_{i,j}$ is always $< h_i$ for job $J_{i,j}$
Output jitter	Output jitter is present and $Out_{i,j}$ can exist within the range of $(0, 2h_i)$ for job $J_{i,j}$
$R_{i,j}$	The response time can vary for job $J_{i,j}$ and is always $< h_i$

Table 5.4: Summary of Configuration 4 Timing Parameters

5.3.5 Comparison of Configurations

Following a discussion of the four configurations, Table 5.5 provides a comparison of these four configurations.

Configuration	Advantages	Disadvantages
Configuration 1	1. Software induced jitter is minimized 2. Lowest worst case processor utilization	1. Adds a long artificial delay between sampling and output 2. Highest utilization of specialized hardware to minimize jitter as much as possible
Configuration 2	Most implemented strategy in the industry	More input-output jitter and output jitter compared to configuration 1
Configuration 3	Output jitter is minimized as much as possible compared to configurations 2 and 4	1. Adds a long artificial delay between sampling and output 2. Most unlikely to be implemented
Configuration 4	Lowest utilization of specialized hardware to minimize jitter; this leads to a reduction in cost to build the digital control system	1. Maximum possible software induced jitter 2. Highest worst case processor utilization

Table 5.5: Comparison of the Four Configurations

Chapter 6

Jitter Compensating PID Controller Solution

The main contribution of this research is a PID controller capable of compensating for software induced jitter. The methodology involved in the design of the proposed solution is elaborated in this Chapter. An overview is initially provided to motivate the design of the proposed solution. Subsequently, the design steps performed in the continuous-time and discrete-time are discussed. An additional contribution made through this research, which involves improving the TrueTime simulator to support a timed hardware trigger is described at the end of this Chapter.

6.1 Design Overview

As introduced in Chapter 3, Section 3.1, solutions that compensate for input-output jitter and output jitter require a prediction based approach. These solutions are referred to as **conventional solutions** in this dissertation. The need for prediction arises due to the fact that traditional solutions from the control systems domain assume a plant is provided with a control signal as soon as the inputs are sampled, which implies a zero-delay execution for the controller. Therefore, prediction involves

delivering a control signal intended to be applied at a pre-determined timing instant in the future. This pre-determined timing instant is taken as the sampling period boundary. Usually, the control signal is predicted after sampling the inputs. Therefore, for the prediction to work it is necessary to eliminate output jitter. However, the presence of sampling jitter in the digital control system is acceptable, and as a consequence input-output jitter will be present. According to the configurations presented in Chapter 5, Section 5.3, implementation strategies outlined in configuration 1 or configuration 3 can only be used to implement digital control systems that use the conventional solution. These two implementation strategies output the control law before the start of the new period to improve performance (see Chapter 7, Section 7.3 for results pertaining to this), which is slightly different in comparison to the conventional solutions. Nevertheless, the periodicity in outputting the control law is maintained as much as possible (the D/A conversion adds a slight amount of jitter).

Contrary to conventional solutions, the proposed jitter compensating PID controller solution can be used in digital control systems implemented using any type of configuration. Performance data shown in Chapter 7, Section 7.3 demonstrates this fact. The only constraint that this solution is bound to is the one sampling period delay assumption.

The remainder of this Chapter is dedicated to a description of the design of the proposed jitter compensating PID controller.

6.2 Continuous-Time Design

Designing the control system along with the PID controller in the continuous-time domain is the initial step of the design methodology. As highlighted in Chapter 2, Section 2.3.1 the MATLAB and Simulink tools are used to automate the design process. The purpose of the continuous-time design is to establish a control system that meets the performance requirements set out by the percent overshoot, rise time,

and settling time, and then use it as a basis for designing the digital control system in the discrete-time domain. After building a simulation model of the continuous-time control system using Simulink, the PID controller block is tuned graphically to produce a control system that meets performance requirements. The rise time in the continuous-time design is required to formulate the appropriate sampling period for the digital control system. This formulation is discussed in Chapter 2, Section 2.1.5, and it is given by,

$$h = \frac{T_r}{N} \quad (6.1)$$

T_r is the rise time and N is an integer with values in the range $[4, 10]$. This research uses $N = 10$ to derive the sampling period h as it is a good compromise between control performance and scheduling multiple tasks on a single processor. The rise time specified in the performance requirements is not used directly to derive the period, but instead, the rise time resulting from the continuous-time control system design that meets all performance requirements is employed. It is expected that the new rise time should be less than or equal to the required rise time. Deriving the sampling period in this manner allows for a digital control system with better performance. This is because, if the resulting rise time is less than the required rise time, then the sampling period will be less. Usually, lowering the sampling period is beneficial for a digital control system. However, choosing an unnecessarily low sampling period places a burden on the software that implements the control task [1].

As highlighted in Chapter 3, Section 3.1.1 the first step in designing a discrete-time delay compensating controller is to include the delay in the process model. This equally applies in the design of PID controllers. In addition, this approach can be used in the continuous-time domain as well. A constant time delay element can be modeled in the continuous-time domain using the Laplace transform as $e^{-s\kappa}$, where κ is the time delay between input sampling to outputting the control signal to the plant.

When using the formula to include a time delay in the process model by Åström et al. [1], which was presented in Chapter 3, Section 3.1.1, the continuous-time control system is modeled so that the delay element is placed between the controller and the plant. The formula to include the delay in the process model is given by (6.2),

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t - \kappa) \quad (6.2)$$

A high-level model of a continuous-time control system with the model of the delay element is depicted in Figure 6.1.

Figure 6.1: Continuous-Time Control System Model with the Delay Element

If Simulink's PID tuner (see Chapter 2, Section 2.3.1) is used to design a delay compensating PID controller then it has to first convert the delay element into a rational transfer function. For continuous-time systems, Simulink uses the Padé approximation and the user has to specify the approximation order [45]. The order n Padé approximation of $e^{-s\kappa}$ is given by (6.3),

$$e^{-s\kappa} \approx \frac{Q(-s\kappa)}{Q(s\kappa)} \quad (6.3)$$

$$Q(s\kappa) = \sum_{k=0}^n \frac{(2n-k)!n!}{2n!k!(n-k)!} (s\kappa)^k$$

Since an approximation is used, it could result in modelling errors. The issue of approximation is overcome in the discrete-time domain which is discussed in Section 6.3.

6.3 Discrete-Time Design

The two methods used to produce a discrete-time controller (e.g. PID controller) are, either designing a continuous-time controller and finally discretizing it, or performing the entire design in the discrete-time domain [38]. This research uses the latter approach to design a digital PID controller. Despite the fact that part of the design is carried out in the continuous-time, this is not the same as former approach, because the continuous-time controller is not discretized. When performing the design, the continuous-time plant and delay element has to be discretized using the sampling period formulated during the continuous-time design phase. Zero-order hold is used as the discretization approach in this research. Furthermore, in this research, the input to output delay κ , introduced in Section 6.2, is equated to the sampling period h_i , for control system i , which is a constant value.

Transforming the delay element of one sampling period to the z -domain results in a delay element of z^{-1} . This transformation is much simpler compared to the Padé approximation in the Laplace domain. The Padé approximation was introduced in Section 6.2. The introduction of a one sampling period delay element results in an additional open-loop pole in the digital control system. The discrete-time version of the the continuous-time control system depicted in Figure 6.1 is shown in Figure 6.2.

Figure 6.2: Discrete-Time Control System with a Delay

After discretizing the one sampling period delay and the plant, the next step is to design the discrete-time PID controller. This is done by using Simulink's PID tuner, which is also used in the continuous-time design in Section 6.2. The discrete-time PID

controller can be designed, using the PID tuner, to satisfy performance requirements as well as to compensate for a delay κ_i up to one sampling period.

When comparing the resulting discrete-time PID controller that compensates for h_i , against a discrete-time PID controller that does not compensate for any delay, the only difference lies in the constants used for the P , I , and D control terms. The underlying simulation model was not updated by Simulink. This implies that the execution time between the two discrete-time PID controllers is equal. Additionally, timing parameters including but not limited to, R_i and the amount of software induced jitter remains the same between the same type of configurations that use and do not use the jitter compensating PID. The unchanged execution time overhead immensely benefits digital control systems with a limited timing budget. The structure of the discrete-time PID controller in the z -domain and the type of approximations used for the integral and derivative components used in this research are discussed in Chapter 2, Section 2.3.1.

The final step of the design is to interface the compensated PID controller with the continuous-time plant through ADCs to sample the plant response and reference signal, and DAC to output the calculated control law to the plant. The modelling of the ADC and DAC used in this research is highlighted in Chapter 5, Section 5.1. Furthermore, the method used to trigger the ADCs and DAC depends on the configuration being used. The configurations are highlighted in Chapter 5, Section 5.3.

6.4 Summary of the Design Process

The methodology to design a discrete-time PID controller to compensate for software induced jitter is discussed in this Chapter. These design steps are summarized in the following list,

1. Continuous-time domain design:

- (a) Design a continuous-time PID controller that satisfies performance requirements (Simulink's PID tuner can be used for this step).
- (b) Obtain the rise time by simulating the continuous-time control system and calculate the sampling period h_i using (6.1).

2. Discrete-time domain design:

- (a) Discretize the plant using the calculated sampling period h_i .
- (b) Model the one sampling period delay of the digital control system in the z -domain and place it between the PID controller and the plant.
- (c) Design the PID controller to compensate for the one sampling period delay and satisfy performance requirements (Simulink's PID tuner can be used for this step).
- (d) Interface the designed discrete-time PID controller with a continuous-time plant using ADCs and DAC.
- (e) Identify that the chosen sampling period, rise time, and settling time can compensate for output jitter and satisfy performance requirements (e.g. via simulation). If performance requirements are violated, then it is recommended to minimize output jitter by using a hardware triggered DAC. Please refer to Chapter 7, Section 7.5 for a detailed discussion of this item.

6.5 Improving TrueTime

As introduced in Chapter 4, Section 4.4, a simulation based performance evaluation is utilized for this research. TrueTime along with Simulink provides a suitable platform to simulate a system executing multiple digital control tasks on the same processor. A limitation of TrueTime is the simulation of triggering the ADC and DAC via hardware. This feature is required to support configurations 1, 2 and 3. As a workaround

it is possible to simulate this behavior through a timer ISR with zero execution time. This approach works well for a system with only one control task. On the other hand, it becomes an issue when executing multiple tasks. This is because the execution of the timer ISR can get delayed due to the execution of a different timer ISR. Consequently, the ADC or DAC will not be triggered at precise instants in time.

An alternative workaround is discussed by Lozoya et al. in [38] where two cascaded TrueTime kernel simulation blocks (see Chapter 2, Section 2.3.2) are used to simulate timed hardware triggered D/A conversion. In their workaround, when the DAC needs to be triggered, a signal is generated from one kernel block and it is received by the second kernel block. An external interrupt is generated on the receiving kernel block, and as a result an ISR is triggered. In the ISR, the DAC is invoked. This workaround also suffers from the issue of being delayed by other ISRs, which was described in this Section.

Therefore, as part of this research a new feature was added to TrueTime to simulate the behavior of triggering the ADC or DAC via hardware when a timer expires. The execution of the triggers are still performed in a timer ISR, however, this timer ISR is not delayed by the execution of other timer ISRs. The timer ISR does not consume any simulation time. If more than one timer expires at the same time all timers are serviced together without a delay between servicing of the timer ISRs. The new feature is not limited to only triggering ADCs and DACs, but it can also be used to simulate any timed hardware trigger. An added benefit is, the timed hardware trigger can be carried out with only one TrueTime kernel block.

The new feature was validated by monitoring the execution of the timed hardware triggered actions via TrueTime. The start times of the hardware triggers were recorded using TrueTime. It was observed that all actions were triggered at precise instants in time without any jitter. Furthermore, by observing TrueTime's task execution monitor, the execution of the timed hardware triggered actions did not interfere with the execution of the software tasks (i.e. no software tasks, ISRs yielded

the processor to execute the timed hardware triggered actions), and no overhead was incurred in its execution.

Adding this new feature entailed modifying the TrueTime kernel to manage a set of timer ISRs, which do not consume execution time and will not be blocked by any other ISR. The management of these ISRs are not combined with regular software timer ISRs. Additionally, the TrueTime application programming interface (API) which defines a timer ISR was also modified. An extra parameter was introduced to the API to differentiate between a software or hardware based timer ISR. The total lines of code required to realize this feature is approximately 35 lines.

Chapter 7

Performance Evaluation

The effectiveness of the proposed solution to compensate for software induced jitter introduced in Chapter 6 is evaluated in this Chapter. A simulation based performance evaluation is carried out where digital control systems that control example plants presented in Chapter 2, Section 2.4 are utilized. The performance of the proposed jitter compensating PID controller is compared against an uncompensated PID controller. Performance results in this Chapter demonstrates that the proposed solution is capable of meeting performance requirements, while the uncompensated PID controller fails to meet one or more performance requirements.

This Chapter presents performance requirements associated with each digital control system example, a detailed treatment of how experiments were set up, performance results, followed by a general discussion of the results. Finally, a limitation of the proposed solution, and a proposed workaround is discussed.

7.1 Performance Requirements

Table 7.1 illustrates the performance requirements and sampling periods used to design discrete-time PID controllers to control the example plants.

Parameter	Hard Disk Servo	Voltage Stabilizer	Industrial Process
h_1	$10\mu s$	3ms	24ms
% Overshoot	$< 10\%$	$< 10\%$	$< 10\%$
Rise time	$< 140\mu s$	$< 30ms$	$< 260ms$
Settling time	$< 580\mu s$	$< 100ms$	$< 850ms$

Table 7.1: Design Parameters for Each Digital Control System Example

Performance requirements for each digital control example were obtained while designing the PID controller. That is, the Simulink PID tuner was used to design an arbitrary PID controller in the continuous-time domain, and as described in Chapter 6, eventually, a discrete-time PID controller was obtained. The step input performance parameters of the arbitrary continuous-time PID controller was used as the basis for formulating the performance requirements used for the digital control system. The design procedure described in Chapter 6 is used to design jitter compensating PID controllers. Uncompensated PID controllers were also designed to control each plant example.

The controller for the hard disk servo digital control system, unlike the other two digital control system examples, is a PD controller. The other two digital control system examples use PID controllers. Since the hard disk servo plant model has a dominant integrator it is essential to use a PD controller to achieve a steady state error of zero. PD controllers are introduced in Chapter 2, Section 2.1.2.

7.2 Experimental Setup

The performance evaluation compares the performance of three types of digital control system sets listed below:

1. Digital control systems implemented using the four configurations with a jitter compensating PID controller; these type of systems are defined as *NewComp*.
2. Digital control systems implemented using the four configurations with an uncompensated PID controller; these type of systems are defined as *UnComp*.
3. Digital control systems that are implemented using configuration 1 and use a one sampling period delay for the input-output delay; these type of systems are defined as *CompOnePeriod*, and are similar to the conventional solution used in the literature to compensate for input-output jitter and output jitter. As discussed in Chapter 6, Section 6.1, conventional solutions found in the literature typically choose the sampling period boundary as the timing reference to output the control signal.

A digital control system set used in the performance evaluation has five independent SISO digital control systems. Using the periodic task model developed in Chapter 2, Section 2.1.5, a digital control system set has a task set: $\{T_1, T_2, T_3, T_4, T_5\}$. Each T_i instantiates an infinite sequence of jobs $J_{i,j}$. However, in reality the sequence of jobs will be finite during the evaluation. Each T_i in the task set has a control law calculation task T_i^C . The relative deadline for job $J_{i,j}^C$ for T_i^C was formulated in Chapter 5, Section 5.2 and it is given by (7.1),

$$D_{i,j}^C = D_i - r_{i,j}^C \quad (7.1)$$

A T_i has an input task T_i^I only if the digital control system set is implemented using configurations 3 or 4. The relative deadline for a job $J_{i,j}^I$ for T_i^I as mentioned in Chapter 5, Section 5.3.3 is initially set to h_i . An initial performance evaluation revealed that none of the jobs in all T_i^C for all three digital control system examples missed their deadlines. However, considering the worst case response time of all jobs in T_i^I a relative deadline of 10% of h_i was applied to each T_i^I .

T_1 implements the workflow of an example digital control system, and as a result, the performance of the digital control system implemented by T_1 is only evaluated. This is sufficient because, all tasks execute periodically in a similar manner. Furthermore, all tasks interfere with each others' execution and no additional random processes are introduced to the digital control system set.

Three digital control system sets are required for the evaluation because three example plants are used. The first digital control system of the digital control system set controls an example plant. During the evaluation, all digital control systems in the digital control system set are implemented using the same configuration.

In each of the three digital control system sets, T_1 always has the longest period and the other four tasks in the task set have shorter periods. Sampling periods used for the five control tasks in each digital control system set, are given in Table 7.2.

Digital control system set	h_1	h_2	h_3	h_4	h_5
Digital control system set with the hard disk servo digital control system	$10\mu s$	$7\mu s$	$6\mu s$	$5\mu s$	$3\mu s$
Digital control system set with the voltage stabilizer digital control system	3.0ms	2.2ms	2.0ms	1.5ms	0.8ms
Digital control system set with the industrial process digital control system	24.0ms	17.5ms	16.1ms	12.0ms	6.0ms

Table 7.2: Control Task Periods Used in Each Digital Control System Set

The control task periods were chosen in such a way that one or more control task periods would be a harmonic of T_1 . This was done to create more interference to T_1 .

Under EDF scheduling, jobs $J_{1,j}^I$ and $J_{1,j}^C$ are preempted more frequently compared to jobs for other tasks. This is because jobs for control tasks with shorter periods are executed more frequently compared to T_1 . Additionally, the execution of ISRs and the RTOS preempts the jobs for T_1 . Frequent preemptions of jobs $J_{1,j}^I$ and $J_{1,j}^C$ imply that there will be more jitter in the start and completion times for these jobs. As a consequence, sampling jitter, input-output jitter, and output jitter are increased. By increasing jitter, the effects of jitter can be better observed.

The performance evaluation used in this research is based upon the performance evaluation done by Buttazzo et al. in [17], and Liu et al. in [75]. The authors also simulated multiple control tasks and only the performance of the first control task was recorded. In addition, the performance evaluation by the authors configured only the first control task to sample the inputs and output the control law. In comparison, the performance evaluation carried out in this research includes the execution of A/D conversion done ISRs, and execution of CSs and the RTOS scheduler. This was done so that the simulation reflects a realistic implementation as much as possible. A single TrueTime kernel was used in this research to simulate all four configurations in one simulation model.

Regardless of the digital control system example, the total processor utilization for a certain configuration remains the same. Therefore, the worst case computational delay for each digital control system example has been scaled according to the sampling period. Table 7.3 illustrates the worst case processor utilization by T_1 and the total processor utilization for each digital control system set. The processor utilizations for configurations 3 and 4 are higher compared to configurations 1 and 2 because of the execution of the input task jobs $J_{i,j}^I$. A higher processor utilization in configuration 2 compared to configuration 1 is a result of triggering the DAC via software. The total processor utilization is kept below 1.0 so that the jobs $J_{i,j}^I$ and $J_{i,j}^C$ of T_i are guaranteed to meet their deadlines when scheduled under EDF.

Configuration	U_{T_1}	U_{T_2}	U_{T_3}	U_{T_4}	U_{T_5}	$\sum U$
Configuration 1	0.275	0.124	0.087	0.12	0.13	0.736
Configuration 2	0.278	0.128	0.091	0.125	0.14	0.762
Configuration 3	0.294	0.15	0.115	0.158	0.205	0.922
Configuration 4	0.296	0.153	0.119	0.163	0.215	0.946

Table 7.3: Processor Utilizations for Each Configuration

Two types of experiments are carried out as part of the performance evaluation. The first experiment involves using a step function as the reference input and four metrics viz. percent overshoot, rise time, settling time, and the IAE cost function are used to measure performance. These metrics are introduced in Chapter 2, Section 2.1.6. The step function never starts at a sampling period boundary for T_1 . Instead, the step function starts within the duration of one sampling period. This start time is varied ten times for a particular digital control system example. The average performance is calculated after varying the start time of the step function for ten times. To exemplify, for the hard disk servo digital control system, the start times of the step function are in the range $(0\mu s, 10\mu s)$. The simulation time for each of the three digital control system sets is confined to 100 sampling periods of T_1 . This simulation time is sufficient because by the end of the simulation, plant response has settled and the steady state error is zero. Ultimately, ten experiments are repeated where each one consists of 100 sampling periods. It is expected that performance data from the UnComp type should violate one or more performance requirements because the uncompensated PID controller is not able to compensate for software induced delays. On the other hand, the NewComp and CompOnePeriod types will satisfy all performance requirements.

The second experiment consists of tracking a square wave and the performance is estimated using an IAE cost function. The period of the square wave is varied nine

times and the average performance is calculated. Each period of the square wave is chosen so that it is a non-harmonic of the sampling period of T_1 . Additionally, half the period of the square wave is approximately equal to the settling time of T_1 . The simulation time for this experiment was set to 1000 sampling periods of T_1 .

The Simulink simulation model of the experiment setup is shown in Figure 7.1, and it is intended to run a step input experiment. In order to run the tracking experiment the reference input needs to be changed to a square wave function. The TrueTime kernel is responsible for generating timer interrupts, triggering the ADCs and DAC, executing the control law, and scheduling multiple control tasks.

Figure 7.1: Simulink Simulation Model

7.3 Experimental Results

Experimental results for the experiments outlined in Section 7.2 are presented in this Section. Note that the performance is only recorded for the first digital control system in each digital control system set. All average performance results are accompanied with their 95% confidence intervals to show the range of values that have a high probability to contain the population average. Abbreviations used in the tables in

this Section are summarized in the following list:

- **Cfg X:** A configuration ranging from 1 to 4.
- **Comp1P:** The CompOnePeriod digital control system set.
- **RT:** Rise time.
- **ST:** Settling time.

7.3.1 Results for the Hard Disk Servo Digital Control System

Performance results for the step input experiment are presented in Table 7.4 and the 95% confidence intervals of these results are shown in Table 7.5. To facilitate the comparison of the data in Table 7.4, Table 7.6 shows the relative performance data between the NewComp and UnComp type systems. The CompOnePeriod type is compared with configuration 1 of the NewComp type. A $\uparrow$ symbol in Table 7.6 indicates an improvement in NewComp over UnComp or CompOnePeriod, and a $\downarrow$ indicates vice versa. Numeric data shown in bold in Table 7.4 indicates a violation of performance requirements.

System Type	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	% Overshoot	6.58	6.07	5.98	5.47
	RT (μ s)	127.28	127.96	128.38	129.37
	ST (μ s)	561.51	555.78	557.3	554.29
	IAE	1.36e-4	1.34e-4	1.34e-4	1.33e-4
UnComp	% Overshoot	11.51	10.88	10.87	10.35
	RT (μ s)	127.6	128.34	128.51	129.34
	ST (μ s)	600.24	593.2	594.68	590.59
	IAE	1.48e-4	1.44e-4	1.45e-4	1.43e-4
Comp1P	% Overshoot	8.7685	N/A	N/A	N/A
	RT (μ s)	123.89	N/A	N/A	N/A
	ST (μ s)	573.69	N/A	N/A	N/A
	IAE	1.4249e-4	N/A	N/A	N/A

Table 7.4: Hard Disk Servo Digital Control System Average Step Input Performance

When comparing the data in Table 7.4 to the performance requirements in Table 7.1, the NewComp type satisfied all performance requirements in all four configurations. The CompOnePeriod type also satisfied all performance requirements. The UnComp type was not able to satisfy the percent overshoot and settling time requirements for all four configurations. Therefore, the proposed jitter compensating PD controller successfully satisfied performance requirements for the hard disk servo digital control system.

System Type	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	% Overshoot	6.58,6.58	6.07,6.07	5.97,5.98	5.36,5.59
	RT (μs)	127.28, 127.28	127.96, 127.96	128.37, 128.39	129.01, 129.73
	ST (μs)	559.44, 563.58	553.71, 557.85	555.14, 559.47	552.70, 555.89
	IAE	1.34e-4, 1.38e-4	1.31e-4, 1.36e-4	1.32e-4, 1.36e-4	1.31e-4, 1.34e-4
UnComp	% Overshoot	11.51, 11.51	10.88, 10.88	10.87, 10.87	10.28, 10.41
	RT (μs)	127.60, 127.60	128.34, 128.34	128.51 128.52	129.14 129.53
	ST (μs)	598.17, 602.31	591.13, 595.27	592.52, 596.85	588.83, 592.33
	IAE	1.46e-4, 1.50e-4	1.42e-4, 1.46e-4	1.43e-4, 1.47e-4	1.41e-4, 1.45e-4
Comp1P	% Overshoot	8.77,8.77	N/A	N/A	N/A
	RT (μs)	123.89,123.89	N/A	N/A	N/A
	ST (μs)	571.62,571.62	N/A	N/A	N/A
	IAE	1.40e-4,1.44e-4	N/A	N/A	N/A

Table 7.5: Hard Disk Servo Digital Control System Step Input Performance 95% Confidence Intervals

Some metrics for some configurations have a zero length confidence intervals mainly because the performance results did not vary. Therefore, the average is also equal the bounds of the confidence interval. The percent overshoot and rise time metrics for configurations 1 and 2 showed this behavior. In addition, when comparing the confidence intervals for a particular metric among the three digital control system types and the same configurations, the percent overshoot, settling time, and

IAE metrics have non overlapping intervals. This implies that there is a statistically significant difference in performance among the three digital control system set types. However, the rise time confidence interval of the CompOnePeriod type does not overlap with configuration 1 for both NewComp and UnComp types. This too infers that the difference in the rise time of CompOnePeriod compared to the other two types is statistically significant. When comparing the confidence intervals for the IAE performance results among the four configurations for NewComp and UnComp, there is overlap. Therefore, it is not possible to conclude that the difference in the performance results among NewComp and UnComp are statistically significant.

Comparison	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp vs. UnComp	% Overshoot	42.8% ↑	44.2% ↑	45.1% ↑	47.1% ↑
	RT	0.3% ↑	0.3% ↑	0.1% ↑	0.02% ↓
	ST	6.5% ↑	6.3% ↑	6.3% ↑	6.2% ↑
	IAE	7.7% ↑	7.6% ↑	7.5% ↑	7.3% ↑
NewComp vs. Comp1P	% Overshoot	24.9% ↑	N/A	N/A	N/A
	RT	2.7% ↓	N/A	N/A	N/A
	ST	2.1% ↑	N/A	N/A	N/A
	IAE	4.3% ↑	N/A	N/A	N/A

Table 7.6: Step Input Relative Performance Data of System Types Which Use the Hard Disk Servo Digital Control System

The relative performance data in Table 7.6 illustrates that the NewComp performed better compared to the UnComp and CompOnePeriod types. However, UnComp performed better by 0.02% in configuration 4 and CompOnePeriod performed better by 2.7% compared to NewComp for the rise time metric. On the other hand, relative data from the IAE metric demonstrates a comprehensive improvement in the NewComp type compared to the UnComp and CompOnePeriod types by at most 7.7%, despite the decline in the rise time compared to CompOnePeriod and UnComp

types. The improvement in performance of the NewComp type compared to the other two types is also confirmed by the non overlapping IAE confidence intervals.

Performance results for the tracking experiment are presented in Table 7.7 and the 95% confidence intervals of these results are provided in Table 7.8. Relative performance data which compares the NewComp against UnComp and CompOnePeriod are highlighted in Table 7.9.

System Type	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	4.25e-3	4.19e-3	4.16e-3	4.12e-3
UnComp	4.59e-3	4.52e-3	4.48e-3	4.43e-3
Comp1P	4.44e-3	N/A	N/A	N/A

Table 7.7: Hard Disk Servo Digital Control System Average Tracking Performance

System Type	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	4.22e-3,4.28e-3	4.16e-3,4.23e-3	4.13e-3,4.19e-3	4.08e-3,4.15e-3
UnComp	4.56e-3,4.62e-3	4.48e-3,4.55e-3	4.45e-3,4.51e-3	4.39e-3,4.47e-3
Comp1P	4.41e-3,4.47e-3	N/A	N/A	N/A

Table 7.8: Hard Disk Servo Digital Control System Tracking Performance 95% Confidence Intervals

The confidence interval for the same configuration among the three different digital control system set types do not overlap, which means that the difference in performance is statistically significant. However, there is overlap among the confidence intervals for all four configurations for NewComp and UnComp types. Therefore, it is not possible to conclude that the difference in performance for this case is statistically significant.

Comparison	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp vs. UnComp	7.36% ↑	7.16% ↑	7.28% ↑	7.13% ↑
NewComp vs. Comp1P	4.28% ↑	N/A	N/A	N/A

Table 7.9: Tracking Relative Performance of System Types Which use the Hard Disk Servo Digital Control System

Performance results both from Table 7.7 and Table 7.9 exemplify that NewComp has better tracking performance compared to UnComp and CompOnePeriod types. This is also confirmed by the confidence interval results in Table 7.8.

7.3.2 Results for the Voltage Stabilizer Digital Control System

Performance results for the step input experiment are presented in Table 7.10 and the 95% confidence intervals for these results are given in Table 7.11.

System Type	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	% Overshoot	4.098	2.9216	2.8684	2.1869
	RT (ms)	21.179	21.664	21.772	22.143
	ST (ms)	87.194	87.746	85.745	85.287
	IAE	0.02026	0.01960	0.01984	0.01961
UnComp	% Overshoot	12.521	11.328	11.572	10.961
	RT (ms)	26.499	26.99	26.952	27.236
	ST (ms)	93.896	93.081	93.991	93.893
	IAE	0.02816	0.02716	0.02756	0.02716
Comp1P	% Overshoot	7.688	N/A	N/A	N/A
	RT (ms)	19.953	N/A	N/A	N/A
	ST (ms)	91.332	N/A	N/A	N/A
	IAE	0.02181	N/A	N/A	N/A

Table 7.10: Voltage Stabilizer Digital Control System Average Step Input Performance

Step input performance results from the NewComp type compared to performance requirements in Table 7.1 exemplify that performance requirements have been satisfied for all four configurations. In contrast, the UnComp type was not able to satisfy only the percent overshoot requirement for all four configurations, but it was able to satisfy the rise time and settling time requirements. The CompOnePeriod type was also successful in satisfying all performance requirements.

System Type	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	% Overshoot	4.098, 4.098	2.922, 2.922	2.858, 2.879	2.178, 2.196
	RT (ms)	21.179, 21.179	21.664, 21.664	21.767, 21.777	22.093, 22.193
	ST (ms)	86.681, 87.707	87.233, 88.259	85.067, 86.423	83.479, 87.095
	IAE	0.01974, 0.02077	0.01909, 0.02011	0.01916, 0.02051	0.01906, 0.02016
UnComp	% Overshoot	12.521 , 12.521	11.328 , 11.328	11.567 , 11.577	10.857 , 11.066
	RT (ms)	26.499, 26.499	26.99, 26.99	26.946, 26.957	27.207, 27.265
	ST (ms)	93.383, 94.409	92.568, 93.594	93.322, 94.658	93.502, 94.283
	IAE	0.02765, 0.02868	0.02666, 0.02768	0.02688, 0.02823	0.02662, 0.02769
Comp1P	% Overshoot	7.688,7.688	N/A	N/A	N/A
	RT (ms)	19.953,19.953	N/A	N/A	N/A
	ST (ms)	90.819,91.845	N/A	N/A	N/A
	IAE	0.02129,0.02232	N/A	N/A	N/A

Table 7.11: Voltage Stabilizer Digital Control System Step Input Performance 95% Confidence Intervals

The confidence interval of each metric for the same configuration among the three different digital control system set types are different and this implies that this difference in performance is statistically significant. Conversely, there is overlap in the confidence intervals for the IAE metric among all four configurations for NewComp or UnComp. This suggests that it is not possible to draw conclusions on the statistical significance of the difference in performance.

Comparison	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp vs. UnComp	% Overshoot	67.3% ↑	74.2% ↑	75.2% ↑	80.0% ↑
	RT	20.1% ↑	19.7% ↑	19.2% ↑	18.7% ↑
	ST	7.1% ↑	5.7% ↑	8.8% ↑	9.2% ↑
	IAE	28.1% ↑	27.9% ↑	28.0% ↑	27.8% ↑
NewComp vs. Comp1P	% Overshoot	46.7% ↑	N/A	N/A	N/A
	RT	5.8% ↓	N/A	N/A	N/A
	ST	4.5% ↑	N/A	N/A	N/A
	IAE	7.1% ↑	N/A	N/A	N/A

Table 7.12: Step Input Relative Performance Data of System Types Which Use the Voltage Stabilizer Digital Control System

Relative performance data depicted in Table 7.12 indicates that the NewComp type performed comprehensively better than the UnComp type. However, the NewComp type was unable to improve the rise time compared to the CompOnePeriod type, and the decline was 5.8%. Nevertheless, the relative IAE performance illustrates that NewComp performed completely better than both UnComp and CompOnePeriod types, and the improvement was at most 28.1%. This fact is confirmed by the non overlapping confidence intervals for the IAE metric.

Performance results for the tracking experiment are presented in Table 7.13 and the 95% confidence intervals of these results are depicted in Table 7.14. The relative performance data which compares the NewComp type against UnComp and CompOnePeriod types are depicted in Table 7.15.

System Type	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	1.064	1.040	1.032	1.019
UnComp	1.433	1.405	1.398	1.381
Comp1P	1.143	N/A	N/A	N/A

Table 7.13: Voltage Stabilizer Digital Control System Average Tracking Performance

System Type	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	1.049,1.079	1.026,1.055	1.021,1.044	1.009,1.030
UnComp	1.418,1.448	1.390,1.420	1.386,1.410	1.369,1.392
Comp1P	1.128,1.158	N/A	N/A	N/A

Table 7.14: Voltage Stabilizer Digital Control System Tracking Performance 95% Confidence Intervals

The confidence intervals for the tracking experiment are similar to the confidence intervals from the step input experiment in the sense that the intervals for the same configuration among the three different digital control system set types do not overlap. This means that the difference in performance is statistically significant. On the other hand, there is overlap among the four configurations for both NewComp and UnComp. This implies that it is not possible to conclude that the difference in performance for this situation is statistically significant.

Comparison	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp vs. UnComp	25.78% ↑	25.95% ↑	26.16% ↑	26.17% ↑
NewComp vs. Comp1P	6.93% ↑	N/A	N/A	N/A

Table 7.15: Tracking Relative Performance of System Types Which use the Voltage Stabilizer Digital Control System

Both data from Table 7.13 and Table 7.15 indicate that the tracking performance of the NewComp type performed better than UnComp and CompOnePeriod types. This fact is confirmed by the non overlapping confidence intervals in Table 7.14. The overlapping confidence intervals among all four configurations for NewComp and UnComp signify that it is not possible to draw conclusions on the statistical significance in the difference in performance.

7.3.3 Results for the Industrial Process Digital Control System

The performance results for the step input experiment are presented in Table 7.16 and the 95% confidence intervals of these results are provided in Table 7.17.

System Type	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	% Overshoot	3.9874	4.837	3.4629	3.6379
	RT (ms)	219.71	205.79	223.46	221.78
	ST (ms)	641.04	617.96	626.71	623.18
	IAE	0.21047	0.20965	0.20574	0.20903
UnComp	% Overshoot	11.718	11.513	10.942	10.322
	RT (ms)	198.7	195.52	200.97	202.21
	ST (ms)	711.41	692.73	707.18	704.18
	IAE	0.23555	0.23181	0.22936	0.22711
Comp1P	% Overshoot	6.4636	N/A	N/A	N/A
	RT (ms)	208.08	N/A	N/A	N/A
	ST (ms)	675.27	N/A	N/A	N/A
	IAE	0.22221	N/A	N/A	N/A

Table 7.16: Industrial Process Digital Control System Average Step Input Performance

The NewComp type that has the industrial process digital control system, similar to the other two digital control system examples, successfully satisfied all performance requirements. This is consistent in all four configurations. The CompOnePeriod also satisfied all performance requirements. On the other hand, the UnComp type was not able to satisfy the percent overshoot requirement, but it was able to satisfy the settling time and rise time requirements.

System Type	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	% Overshoot	3.987, 3.987	4.837, 4.837	3.460, 3.466	3.045, 4.2304
	RT (ms)	219.71, 219.71	205.79, 205.79	223.4, 223.51	211.39, 232.17
	ST (ms)	635.93, 646.15	612.85, 623.07	621.4, 632.02	619.11, 627.26
	IAE	0.20536, 0.21558	0.20454, 0.21476	0.20047, 0.21101	0.20588, 0.21218
UnComp	% Overshoot	11.718 , 11.718	11.513 , 11.513	10.935 , 10.949	9.925, 10.719
	RT (ms)	198.7, 198.7	195.52, 195.52	200.95, 200.99	198.9, 205.51
	ST (ms)	706.3, 716.52	687.62, 697.84	701.91, 712.45	697.44, 710.93
	IAE	0.2304, 0.2407	0.2267, 0.2369	0.2241, 0.2347	0.2223, 0.2319
Comp1P	% Overshoot	6.464,6.464	N/A	N/A	N/A
	RT (ms)	208.08,208.08	N/A	N/A	N/A
	ST (ms)	670.164,680.376	N/A	N/A	N/A
	IAE	0.2171,0.2273	N/A	N/A	N/A

Table 7.17: Industrial Process Digital Control System Step Input Performance 95% Confidence Intervals

The confidence interval for each metric for the same configuration among the three digital control system set types does not overlap. This is also observed in the voltage stabilizer digital control system performance results. The non overlapping confidence interval implies that the difference in performance is statistically significant. In contrast, there is overlap in the confidence intervals among all four configurations for NewComp and UnComp. Therefore, for this situation, the statistical significance of the difference in performance cannot be inferred.

Comparison	Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp vs. UnComp	% Overshoot	66.0% ↑	58.0% ↑	68.4% ↑	64.8% ↑
	RT	9.6% ↓	5.0% ↓	10.1% ↓	8.8% ↓
	ST	9.9% ↑	10.8% ↑	11.4% ↑	11.5% ↑
	IAE	10.6% ↑	9.6% ↑	10.3% ↑	8.0% ↑
NewComp vs. Comp1P	% Overshoot	38.3% ↑	N/A	N/A	N/A
	RT	5.3% ↓	N/A	N/A	N/A
	ST	5.1% ↑	N/A	N/A	N/A
	IAE	5.3% ↑	N/A	N/A	N/A

Table 7.18: Step Input Relative Performance Data of System Types Which Use the Industrial Process Digital Control System

Data from the relative performance indicates that the NewComp type improved the percent overshoot and the settling time. However, the NewComp type was not able to improve the rise time in all four configurations. Likewise, the NewComp type improved the percent overshoot and settling time compared to the CompOnePeriod type, but, it could not improve the rise time. Nevertheless, the relative data from the IAE metric indicates that the NewComp type performed comprehensively better than the UnComp and CompOnePeriod types. An improvement of at most 10.6% was observed from the NewComp type compared to the UnComp and CompOnePeriod types. The improvement is also confirmed by the non overlapping confidence intervals in Table 7.17.

Performance results for tracking experiment are presented in Table 7.19 and the 95% confidence intervals of these results are highlighted in Table 7.20. Relative performance data which compares the NewComp type against the UnComp and CompOnePeriod types are illustrated in Table 7.21.

System Type	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	11.715	11.599	11.617	11.594
UnComp	12.862	12.618	12.715	12.544
Comp1P	12.307	N/A	N/A	N/A

Table 7.19: Industrial Process Digital Control System Average Tracking Performance

System Type	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp	11.527,11.902	11.395,11.802	11.484,11.750	11.420,11.769
UnComp	12.677,13.047	12.429,12.807	12.585,12.846	12.395,12.692
Comp1P	12.117,12.496	N/A	N/A	N/A

Table 7.20: Industrial Process Digital Control System Tracking Performance 95% Confidence Intervals

The non overlapping confidence interval for the same configuration among the three digital control system types implies that the difference in performance is statistically significant. This is also observed in the step input experiment. However, there is overlap among the four configurations for NewComp and UnComp. Therefore, in this case, it is not possible to conclude that the difference in performance is statistically significant.

Comparison	Cfg 1	Cfg 2	Cfg 3	Cfg 4
NewComp vs. UnComp	9.0% ↑	8.1% ↑	8.6% ↑	7.6% ↑
NewComp vs. Comp1P	4.8% ↑	N/A	N/A	N/A

Table 7.21: Tracking Relative Performance of System Types Which use the Industrial Process Digital Control System

Tracking experimental results and tracking relative performance data for the NewComp type that has the industrial process digital control system also agrees with data from the other two example digital control system examples in the sense that, the NewComp type has better performance over the UnComp and CompOnePeriod types. This fact is confirmed by the non overlapping confidence intervals in Table 7.20.

7.4 General Discussion of the Experimental Results

Experimental results for the step input experiment shown in Tables 7.4, 7.10, and 7.16 illustrate that the three digital control system examples that used the proposed jitter compensating PID controller are able to satisfy the percent overshoot, rise time, and settling time performance requirements in a consistent manner. Therefore, the three digital control system examples successfully demonstrated the effectiveness of the proposed solution to mitigate the performance degradation effects of software induced jitter.

Out of the three performance metrics, the percent overshoot was mostly affected by the presence of software induced jitter compared to the other two metrics. The NewComp type improved the percent overshoot by at most 80% for configuration 4 for the voltage stabilizer digital control system.

The relative performance data for the step input experiment illustrate that the NewComp type performed better compared to the UnComp and CompOnePeriod types. Nevertheless, the NewComp type was not able to improve the rise time consistently for all three digital control systems compared to the Uncomp and CompOnePeriod types. The CompOnePeriod was better by at most 5.8% and the UnComp type was better at most 10% for the rise time metric. Additionally, the NewComp type for the industrial process digital control system could not improve the rise time compared

to the UnComp type. To the best of our knowledge, a proper explanation for this phenomenon was not found in the literature. In spite of the decline in the rise time, the relative performance data from the IAE cost function illustrated that the NewComp type outperformed the UnComp and CompOnePeriod types. The best overall improvement was at most 28.1%. The non overlapping 95% confidence interval results for the IAE metric for all three digital control system examples and three digital control system set types indicates that the difference in performance is statistically significant.

A conclusion that can be drawn by examining the data is the large input to output delay in the CompOnePeriod type compared to configuration 1 of the NewComp type is capable of improving the rise time, albeit a decline in the percent overshoot and settling time. On the contrary, having a large input to output delay usually results in a poor design and this was pointed out in [1], [17], and discussed in Chapter 4, Section 4.1.

Performance results from the tracking experiment can also be used to obtain a complete picture of the performance of the three system types (NewComp, UnComp, and CompOnePeriod). However, as mentioned in Chapter 2, Section 2.1.6 a cost function does not indicate if performance requirements have been met or not. Therefore, it is important to also rely on performance data from a step input experiment. Relative performance data from all three digital control system examples indicate that the NewComp type performs better compared to the UnComp and CompOnePeriod types. The non overlapping 95% confidence intervals confirmed that there is a statistically significant difference in performance among the three digital control system set types for all three digital control system examples. The best improvement was from the voltage stabilizer digital control system where the NewComp improved the performance by 26.17% compared to the UnComp type for configuration 4.

A conclusion could not be drawn by examining the performance results to determine which configuration would be the best candidate to implement a digital control

system. This is further complicated by the overlapping confidence intervals for the four configurations for NewComp and UnComp. From an implementation point of view, configurations 3 and 4 appear costly, because of the addition of a task that samples the inputs by triggering the ADCs, which can be considered to be redundant. Contemporary microcontrollers incorporate ADCs with a timed hardware trigger. Therefore, this facility can be used to eliminate sampling jitter. Between configurations 1 and 2, configuration 2 would appear to be a better implementation strategy, because according to tracking performance results configuration 2 usually performed better compared to configuration 1 except for the industrial process digital control system. Therefore, merely eliminating input-output jitter and output jitter by adding a constant artificial delay as in configuration 1 is not a preferred implementation strategy for digital control systems. This conclusion was also reached by Buttazzo et al. in [17] and Åström et al. in [1].

7.5 Limitations of the Proposed Solution

While evaluating the performance of the proposed solution, a limitation of the proposed solution was identified. That is, for certain combinations of the sampling period, rise time and settling time, the solution is not capable of compensating for output jitter. This manifests in failing to satisfy performance requirements. The limitation was discovered when compensating for output jitter in configurations 2 and 4. However, this issue was not observed in configurations 1 and 3. The contribution by sampling jitter towards the issue was ruled out by observing that configuration 3 does not exhibit the limitation. Additionally, since configuration 3 has input-output jitter in spite of minimizing output jitter, it is argued that input-output jitter solely does not add to the limitation.

The combination of sampling period, rise time and settling time can be found iteratively by decreasing the rise time by a factor of 10. This also decreases the

settling time. According to the sampling period calculation equation (2.3) introduced in Chapter 2, Section 2.1.5, a decrease in the rise time implies a shorter sampling period is necessary. The iterative process is stopped once there is a violation of at least one performance requirement. After reducing the sampling period, the digital control system is designed and evaluated to determine if performance requirements are violated or not.

To demonstrate this limitation the digital control system set with the voltage stabilizer digital control system is used. The rise time obtained after tuning the jitter compensating PID controller used in the initial performance evaluation is 21ms. The iterative process mentioned in this Section revealed that the rise time and sampling period should be reduced by a factor of 100 to observe a violation. Therefore, the rise time and sampling period was lowered to $120\mu\text{s}$ (from 21ms) and $30\mu\text{s}$ (from 3ms) respectively. In this manner a faster responding digital control system with the jitter compensating PID controller was designed. The processor utilization was not changed from its initial value in this experiment, but instead, all control task periods, ADC and DAC and software execution times were also scaled down by two orders of magnitude. Therefore, to flexibly reduce the execution times it was assumed that the underlying hardware platform can be changed to accommodate the execution requirements.

The step input experiment as described in Section 7.2 was repeated for the digital control system set. The simulation time used for the experiment was 11000 h_1 periods. This simulation time is much higher than what was used in the original step input experiment described in Section 7.2, which was 100 h_1 periods. Since the sampling period for the voltage stabilizer digital control system was lowered by a factor of 100, sampling periods for the other four control tasks were also lowered by the same order of magnitude. Furthermore, since the new digital control system has a faster response, the performance requirements were also revised. The performance requirements for the new voltage stabilizer digital control system is shown in Table 7.22.

Parameter	Requirement
% Overshoot	< 10%
Rise time (μs)	< 140
Settling time (μs)	< 510

Table 7.22: Performance Requirements for the Voltage Stabilizer Digital Control System with a Faster Response

Performance results obtained after executing the step input experiment are shown in Table 7.23. Bold font entries in the Table indicates a violation of a performance requirement.

Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
% Overshoot	6.74	12.44	4.139	7.62
RT (μs)	136.06	128.42	144.31	153.13
ST (μs)	495.58	Didn't reach settling threshold	482.48	Didn't reach settling threshold

Table 7.23: Average Step Input Performance for Voltage Stabilizer Digital Control System with a Faster Response

The 95% confidence intervals of the results in Table 7.23 are shown in Table 7.24. The confidence intervals for the settling time for configurations 2 and 4 are not available because there was not any results for the confidence interval calculation.

Metric	Cfg 1	Cfg 2	Cfg 3	Cfg 4
% Overshoot	6.74,	11.53,	4.13,	4.75,
	6.74	13.36	4.15	10.49
RT (μs)	136.06,	126.55,	144.05,	132.03
	136.06	130.29	144.58	174.21
ST (μs)	490.45,	Didn't reach	475.85,	Didn't reach
	500.71	settling threshold	489.10	settling threshold

Table 7.24: 95% Confidence Intervals for Step Input Performance for Voltage Stabilizer Digital Control System with a Faster Response

Performance results in Table 7.23 highlight the fact that configurations 2 and 4 with the proposed jitter compensating PID controller are not able to satisfy the rise time and settling time, and percent overshoot and settling time requirements respectively. On the other hand, configurations 1 and 3 are able to satisfy all performance requirements.

Both bounds of the confidence interval for metrics that violate performance requirements also violate performance requirements. Therefore a performance violation will also be observed in the population average performance. Furthermore, the upper bound of the percent overshoot confidence interval for configuration 4 violated performance requirements. This implies that it is possible to observe a performance requirement violation for the percent overshoot for configuration 4 in a different situation.

MATLAB normally uses a $\pm 2\%$ threshold when measuring the settling time. That is, when the plant response is settling, the settling time is recorded when the response crosses and remains in the $\pm 2\%$ threshold. The final value of the step input used in this research is 1, so the $\pm 2\%$ threshold is at ± 1.02 . Therefore, the *Didn't reach settling threshold* in Table 7.23 refers to the fact that the plant response did not pass and

remain in the $\pm 2\%$ threshold after 11000 h_1 periods. Nevertheless, the plant response for both configurations 2 and 4 reached a value of 1.025 and 1.03 respectively.

A proposed workaround for the issue of not meeting the performance requirements in configurations 2 and 4 is to minimize the output jitter as much as possible. In other words, this means to adopt configurations 1 and 3. Jitter from the A/D and D/A conversions contribute to output jitter in configurations 1 and 3. However, this jitter is not large compared to what is observed in configurations 2 and 4. Unfortunately, this research was unable to identify an analytical approach to determine the rise time, settling time and sampling period combination that results in failing to satisfy the performance requirements in configurations 2 and 4. Section 7.4 concluded that from an implementation point of view configuration 2 would be a good choice. In spite of that, if configuration 2 appears to not satisfy performance requirements due to the limitation, then falling back to configuration 1 is recommended.

Chapter 8

Conclusions and Future Research

Satisfying performance requirements in a digital control system is a concern on par with ensuring stability. There are many aspects that can affect the performance of a digital control system and this research focused on the negative effects of software induced delays. Undesirable effects entail a performance degradation where performance requirements may not be satisfied. In the worse case the performance degradation could lead to instability.

The thesis in this dissertation proposed a solution to mitigate the performance degradation due to software induced sampling jitter, input-output jitter, and output jitter. Application of the proposed solution to example digital control systems found in the literature demonstrated its effectiveness in improving performance.

The solution involves designing a PID controller that compensates for a constant one sampling period delay. The model of the design places a constant one sampling period delay between the controller and plant. The resulting PID controller is also capable of compensating for jitter. The jitter compensating PID controller has exactly the same implementation as its uncompensated counterpart with the exception of updated constants for the P , I , and D terms. This implies that both the compensating and non-compensating PID controllers have the same execution time. The unchanged execution time is a huge benefit for systems where scheduling multiple software tasks

imposes an issue due to high processor utilization. The proposed solution does not add a burden to systems where execution time is a premium.

In addition, this solution does not require a model of the plant for the compensation unlike in state feedback control, which in turn reduces the implementation complexity. The proposed jitter compensating PID controller can be used in many types of digital control system implementations. The types of digital control system implementations considered in this research are described in Chapter 5.

A simulation based evaluation is utilized to compare the performance of three types of digital control system sets: NewComp, UnComp and CompOnePeriod. The CompOnePeriod type is similar to conventional solutions because it also outputs the control signal at a fixed timing reference, which is usually taken as the start of the next sampling period. Two types of experiments are employed where the first experiment measures the performance of a digital control system in response to a step input and the second experiment measures the tracking performance.

Performance results exemplified that the NewComp type satisfied performance requirements under software induced jitter in response to the step input. In contrast, the UnComp type did not satisfy one or more performance requirements. The CompOnePeriod type also met all performance requirements.

Relative performance data in Tables 7.6, 7.12, and 7.18 in Chapter 7, Section 7.3, illustrates that the NewComp type performed better than the UnComp and CompOnePeriod types. The percent overshoot had the best overall improvement of at most 80% for the industrial process digital control system compared to the UnComp type. However, the rise time was not comprehensively improved. The decline in the rise time was at most 10% for the NewComp type that has the industrial process digital control system compared to the UnComp type. Despite the decline in the rise time, relative performance data from the IAE cost function demonstrates an overall improvement in the NewComp type over both the UnComp and CompOnePeriod

types, and the difference in performance is statistically significant. The best improvement was 28.1 % for the NewComp type that has the voltage stabilizer digital control system compared to the UnComp type.

Tracking performance also substantiated that the NewComp type out performed the UnComp and CompOnePeriod types, and the difference in performance is statistically significant. According to Tables 7.9, 7.15, and 7.21 in Chapter 7, Section 7.3, the NewComp type that has the industrial process digital control system improved the performance by at most 68% compared to the UnComp type.

The research identified a limitation of the proposed solution in which the delay compensating PID controller failed to compensate for output jitter for certain combinations of sampling period, rise time, and settling time. This limitation is observed when the digital control system fails to satisfy one or more performance requirements. However, the research was unable to produce an analytical method to determine the failure point and it was discovered through experimentation. Typically as the digital control system responds quickly, that is, decreasing the rise time, settling time, and sampling period beyond a certain threshold produces the failure. A method proposed through this research to fix this situation is to minimize output jitter by triggering the DAC via hardware.

Experimental results were not comprehensive enough to judge which configuration had the overall best performance. This fact is made worse by the non overlapping confidence interval data among all four configurations for NewComp and UnComp. However, from an implementation point of view configuration 2 was identified to be the best candidate. This is because contemporary microcontrollers incorporate ADCs that can be triggered continuously via hardware and therefore, are used to sample inputs with minimal jitter [81], [82]. In the event that configuration 2 fails to satisfy performance requirements due to the limitation identified in this research, it is recommended to utilize configuration 1.

8.1 Summary of Contributions

The focus of this research is to address the problem of performance degradation in digital control systems due to software induced jitter. To this end, two contributions were made, which are summarized below,

1. The solution proposed in this research is a software induced jitter compensating PID controller. The automated PID tuner by MATLAB and Simulink are used in the design of the solution. The proposed jitter compensating PID controller was applied to three digital control system examples found in the literature and a performance evaluation demonstrated that digital control systems that incorporated the proposed PID controller were able to satisfy performance requirements successfully.
2. A new feature was implemented for the TrueTime simulator to simulate a timed hardware triggered action. The need for this feature came to light when attempting to trigger ADCs and DACs at precise instants in time for configurations 1, 2 and 3. The end result is elimination of sampling jitter and minimizing input-output jitter, and output jitter as much as possible. Variations in the execution times of the D/A conversion cause the remaining input-output jitter and output jitter.

8.2 Recommendations for Future Research

This research demonstrated that a digital control system with a jitter compensating PID controller was successful in meeting performance requirements in the presence of software induced jitter. Nevertheless, there are several possible avenues for further research. Recommendations for further research are presented as follows:

1. An automated method to design jitter compensating PID controllers has its merit over a manual approach as used in this research. The design automation

should take performance requirements into consideration and at the same time use a cost function to optimize the performance. A quadratic cost function used to design LQG controllers may not be required, but instead, the optimization can be based on the IAE cost function [20]. A search over the available design space would yield the best performing PID controller.

2. A different direction that can be taken is to implement the proposed solution for an actual digital control system. This research can be used as a basis to identify how well the solution integrates with measures taken when implementing an actual digital control system. Practical issues include but are not limited to, the A/D quantization and round-off effects, plant saturation, plant dead-time, and aliasing of input signals. These issues are briefly discussed in Chapter 2, Section 2.2.
3. Another research opportunity is to utilize this solution to develop a control scheduling co-design approach. That is, an iterative approach can be used to design a digital control system set that maximizes the performance of all of the digital control systems. The iterative approach can leverage the recommendation in item 1 to automate the design of the jitter compensating PID controller to maximize performance. Furthermore, sampling periods and relative deadlines can be optimized to improve the performance even more. A possible workflow for the iterative approach is to initially pick a sampling period based on the requirements and then optimize the relative deadline and determine if the software tasks are schedulable. Then attempt to optimize the performance of the jitter compensating PID controller. This workflow can be repeated until there is no further improvement.
4. The proposed solution in this research was developed under the proposition that the controlled plants are linear in nature and the digital control systems are SISO. However, the majority of the plants in the real world are nonlinear,

and some types of feedback control require a MIMO solution. Therefore, it is worthwhile to develop jitter compensating PID controllers for these complex problems as well.

References

- [1] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems*, 3rd ed. Mineola, NY: Dover Publications, January 2011.
- [2] A. Gambier, “Real-time control systems: a tutorial,” in *Control Conference, 2004. 5th Asian*, vol. 2, Melbourne, Australia, July 2004, pp. 1024–1031.
- [3] K. Åström and T. Häggglund, *PID Controllers: Theory, Design, and Tuning*, 2nd ed. Research Triangle Park, NC: The Instrument Society of America, 1995.
- [4] K. Ogata, *Modern Control Engineering*, 4th ed. Upper Saddle River, NJ: Prentice Hall, 2002.
- [5] N. Nise, *Control Systems Engineering*, 4th ed. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2004.
- [6] K. Karagozyan, “AN-227: Digital control system design with the ADSP-2100 family,” Application Note, 2008.
- [7] C. Schmid, “The classical three-term PID controller,” <http://virtual.cvut.cz/course/syscontrol/node61.html>, [Accessed: November 2013].
- [8] K. J. Åström and R. M. Murray, *Feedback Systems, An Introduction for Scientists and Engineers*. Princeton University Press, Princeton and Oxford, January 2008.

- [9] K. R. Varmah, *Control Systems*. New Delhi, India: Tata McGraw-Hill, July 2010.
- [10] R. Büchi, *State Space Control, LQR and Observer*. Norderstedt, Germany: Books on Demand, July 2010.
- [11] L. Arturo, *Optimal Control An Introduction*. Basel, Switzerland: Birkhäuser Verlag, 2001.
- [12] T.-C. Yang, “Networked control system: a brief survey,” *Control Theory and Applications, IEE Proceedings -*, vol. 153, no. 4, pp. 403–412, 2006.
- [13] W. Zhao, B. H. Kim, A. Larson, and R. Voyles, “FPGA implementation of closed-loop control system for small-scale robot,” in *Advanced Robotics, 2005. ICAR '05. Proceedings., 12th International Conference on*, Seattle, WA, USA, July 2005, pp. 70–77.
- [14] C. L. Liu and J. W. Layland, “Scheduling algorithms for multiprogramming in a hard-real-time environment,” *J. ACM*, vol. 20, no. 1, pp. 46–61, January 1973.
- [15] G. C. Buttazzo, *Hard Real-Time Computing Systems Predictable Scheduling Algorithms and Applications*, 3rd ed. New York, NY, USA: Springer, 2011.
- [16] J. W. S. Liu, *Real-Time Systems*, 1st ed. Englewood Cliffs, NJ, USA: Prentice Hall, April 2000.
- [17] G. Buttazzo and A. Cervin, “Comparative assessment and evaluation of jitter control methods,” in *Proceedings of the 15th Real-Time and Network Systems Conference on*, Nancy, France, March 2007, pp. 163–172.
- [18] P. Marti, J. Fuertes, G. Fohler, and K. Ramamritham, “Jitter compensation for real-time control systems,” in *Real-Time Systems Symposium, 2001. (RTSS 2001). Proceedings. 22nd IEEE*, London, England, December 2001, pp. 39–48.

- [19] S. Baruah, G. Buttazzo, S. Gorinsky, and G. Lipari, "Scheduling periodic task systems to minimize output jitter," in *Real-Time Computing Systems and Applications, 1999. RTCSA '99. Sixth International Conference on*, Hong Kong, China, 1999, pp. 62–69.
- [20] N. J. Killingsworth and M. Krstic, "PID tuning using extremum seeking: online, model-free performance optimization," *Control Systems Magazine, IEEE*, vol. 26, no. 1, pp. 70–79, February 2006.
- [21] S. Ovaska and O. Vainio, "Predictive compensation of time-varying computing delay on real-time control systems," *IEEE Transactions on Control Systems Technology*, vol. 5, no. 5, pp. 523–526, 1997.
- [22] K. Årzén, A. Cervin, J. Eker, and L. Sha, "An introduction to control and scheduling co-design," in *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, vol. 5, Sydney, Australia, December 2000, pp. 4865–4870.
- [23] Analog Devices, "Analog to digital converters," [Accessed: November 2013]. [Online]. Available: <http://www.analog.com/en/analog-to-digital-converters/ad-converters/products/index.html>
- [24] Texas Instruments, "Products for analog to digital converter," [Accessed: November 2013]. [Online]. Available: <http://www.ti.com/lscs/ti/data-converters/analog-to-digital-converter-products.page>
- [25] B. Wittenmark, K. J. Åström, and K.-E. Årzén, "Computer control: An overview," IFAC Professional Brief, Tech. Rep., January 2002.
- [26] T. Hägglund, "A predictive PI controller for processes with long dead times," *Control Systems, IEEE*, vol. 12, no. 1, pp. 57–60, February 1992.
- [27] Mathworks, "Mathworks - MATLAB and simulink for technical computing," [Accessed: November 2013]. [Online]. Available: <http://www.mathworks.com>

- [28] Mathworks, “PID tuning algorithm for linear plant model,” [Accessed: November 2013]. [Online]. Available: <http://www.mathworks.com/help/control/ref/pid tune.html>
- [29] Mathworks, “Automated tuning of the simulink PID controller block,” [Accessed: November 2013]. [Online]. Available: <http://www.mathworks.com/help/slcontrol/gs/automated-tuning-of-simulink-pid-controller-block.html>
- [30] Mathworks, “PID controller, discrete PID controller,” [Accessed: November 2013]. [Online]. Available: <http://www.mathworks.com/help/simulink/slref/pidcontroller.html>
- [31] K. Ogata, *Discrete-Time Control Systems*, 2nd ed. Englewood Cliffs, NJ: Prentice Hall, 1995.
- [32] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K. Årzén, “How does control timing affect performance? analysis and simulation of timing using Jitterbug and TrueTime,” *IEEE Control Systems Magazine*, vol. 23, no. 3, pp. 16–30, June 2003.
- [33] A. Cervin, D. Henriksson, and M. Ohlin, *TrueTime 2.0 beta Reference Manual*, June 2010.
- [34] T. Goh, Z. Li, B. Chen, T. H. Lee, and T. Huang, “Design and implementation of a hard disk drive servo system using robust and perfect tracking approach,” *IEEE Transactions on Control Systems Technology*, vol. 9, no. 2, pp. 221–233, March 2001.
- [35] B. Isayed and M. Hawwa, “A nonlinear PID control scheme for hard disk drive servosystems,” in *Control Automation, 2007. MED ’07. Mediterranean Conference on*, Athens, Greece, June 2007, pp. 1–6.

- [36] X. Sun, J. Lu, and V. Shaker, “Hard disk servo modeling and control,” Project Report, Concordia University, 2002.
- [37] K. C. Tan, R. Sathikannan, W. W. Tan, and A. P. Loh, “Evolutionary design and implementation of a hard disk drive servo control system,” *Soft Comput.*, vol. 11, no. 2, pp. 131–139, September 2006.
- [38] C. Lozoya, M. Velasco, and P. Marti, “The one-shot task model for robust real-time embedded control systems,” *IEEE Transactions on Industrial Informatics*, vol. 4, no. 3, pp. 164–174, August 2008.
- [39] P. Marti, M. Velasco, J. Fuertes, A. Camacho, and G. Buttazzo, “Design of an embedded control system laboratory experiment,” *IEEE Transactions on Industrial Electronics*, vol. 57, no. 10, pp. 3297–3307, October 2010.
- [40] Mathworks, “ss2tf,” [Accessed: November 2013]. [Online]. Available: <http://www.mathworks.com/help/matlab/ref/ss2tf.html>
- [41] D. Patrick and S. Fardo, *Industrial Process Control Systems*, 2nd ed. Lilburn, GA, USA: The Fairmont Press, Inc., 2009.
- [42] W. Yu, D. I. Wilson, J. Currie, and B. R. Young, “The robustness of PI and PID controllers in the presence of sampling jitter,” in *Chemical Process Control VIII, FOCAPO/CPC 2012*, Savannah, Georgia, USA, January 2012.
- [43] M. Ono, H. Shibasaki, K. Matsumoto, K. Sasaki, and Y. Ishida, “Discrete modified smith predictor based on optimal control method for a plant with an integrator,” in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, Istanbul, Turkey, October 2010, pp. 630–635.
- [44] T. Laakso, V. Valimaki, M. Karjalainen, and U. Laine, “Splitting the unit delay [FIR/all pass filters design],” *Signal Processing Magazine, IEEE*, vol. 13, no. 1, pp. 30–60, January 1996.

- [45] L. Ljung, *System Identification ToolboxTM User's Guide*, 2013.
- [46] Mathworks, "Modeling computational delay and sampling effects," [Accessed: November 2013]. [Online]. Available: <http://www.mathworks.com/help/slcontrol/examples/modeling-computational-delay-and-sampling-effects.html>
- [47] S. Bibian and H. Jin, "Time delay compensation of digital control for dc switch-mode power supplies using prediction techniques," *IEEE Transactions on Power Electronics*, vol. 15, no. 5, pp. 835–842, September 2000.
- [48] C.-Y. Kao and B. Lincoln, "Simple stability criteria for systems with time-varying delays," *Automatica*, vol. 40, no. 8, pp. 1429 – 1434, August 2004.
- [49] A. Cervin, "Stability and worst-case performance analysis of sampled-data control systems with input and output jitter," in *2012 American Control Conference*, Montréal, Canada, June 2012.
- [50] V. Zakian, *Control Systems Design: A New Framework*. London, UK: Springer-Verlag, 2005.
- [51] J. Skaf and S. Boyd, "Analysis and synthesis of state-feedback controllers with timing jitter," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 652–657, March 2009.
- [52] A. Cervin, "Improved scheduling of control tasks," in *Real-Time Systems, 1999. Proceedings of the 11th Euromicro Conference on*, York, England, June 1999, pp. 4–10.
- [53] K. Smeds and X. Lu, "Effect of sampling jitter and control jitter on positioning error in motion control systems," *Precision Engineering*, vol. 36, no. 2, pp. 175–192, April 2012.

- [54] P. Marti, J. Fuertes, and G. Fohler, "Sampling jitter compensation in real-time control systems," in *Work-in-progress Session, 13th Euromicro Real-Time Systems Conference on*, Delft, The Netherlands, June 2001.
- [55] P. Marti, R. Villa, J. Fuertes, and G. Fohler, "Stability of on-line compensated real-time scheduled control tasks," in *IFAC New Technologies for Computer Control Conference on*, Hong Kong, China, November 2001.
- [56] P. Marti, R. Villa, J. Fuertes, and G. Fohler, "On real time control tasks schedulability," in *Proceedings of the European control conference ECC'01*, Porto, Portugal, September 2001, pp. 2227–2232.
- [57] D. Niculae, C. Plaisanu, and D. Bistriceanu, "Sampling jitter compensation for numeric pid controllers," in *Automation, Quality and Testing, Robotics, 2008. AQTR 2008. IEEE International Conference on*, vol. 2, Cluj-Napoca, Romania, 2008, pp. 100–104.
- [58] B. Lincoln, "Jitter compensation in digital control systems," in *American Control Conference, 2002. Proceedings of the 2002*, Anchorage, Alaska, USA, May 2002, pp. 2985–2990.
- [59] J. Nilsson, B. Bernhardsson, and B. Wittenmark, "Stochastic analysis and control of real-time systems with random time delays," *Automatica*, vol. 34, pp. 57–64, January 1998.
- [60] P. Balbastre, I. Ripoll, and A. Crespo, "Minimum deadline calculation for periodic real-time tasks in dynamic priority systems," *IEEE Transactions on Computers*, vol. 57, no. 1, pp. 96–109, January 2008.
- [61] T. Kim, H. Shin, and N. Chang, "Deadline assignment to reduce output jitter of real-time tasks," in *Proceedings of the 16th IFAC workshop on Distributed Computer Control Systems*, Sydney, Australia, November 2000, pp. 67–72.

- [62] H. Hoang and G. Buttazzo, “Reducing delay and jitter in software control systems,” in *Real-Time and Network Systems (RTNS), 2007 15th Conference on*, Nancy, France, March 2007, pp. 173–182.
- [63] P. Zhou and J. Xie, “Online deadline assignment to reduce output jitter of real-time control tasks,” in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*, Guanzhou, China, August 2005, pp. 1312–1315.
- [64] S. Hong, X. Hu, and M. Lemmon, “Reducing delay jitter of real-time control tasks through adaptive deadline adjustments,” in *Real-Time Systems (ECRTS), 2010 22nd Euromicro Conference on*, Brussels, Belgium, July 2010, pp. 229–238.
- [65] I. Bate and A. Burns, “An approach to task attribute assignment for uniprocessor systems,” in *Real-Time Systems, 1999. Proceedings of the 11th Euromicro Conference on*, York, England, June 1999, pp. 46–53.
- [66] A. Crespo, I. Ripoll, and P. Albertos, “Reducing delays in RT control: the control action interval,” in *Proceedings of the 14th IFAC World Congress*, Hong Kong, July 1999.
- [67] P. Albertos, A. Crespo, I. Ripoll, M. Valles, and P. Balbastre, “RT control scheduling to reduce control performance degrading,” in *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, Sydney, Australia, December 2000, pp. 4889–4894.
- [68] P. Balbastre, I. Ripoll, J. Vidal, and A. Crespo, “A task model to reduce control delays,” *Real-Time Systems*, vol. 27, pp. 215–236, September 2004.
- [69] M. Lluesma, A. Cervin, P. Balbastre, I. Ripoll, and A. Crespo, “Jitter evaluation of real-time control systems,” in *Embedded and Real-Time Computing Systems*

- and Applications, 2006. Proceedings. 12th IEEE International Conference on*, Sydney, Australia, August 2006, pp. 257–260.
- [70] M. D. Natale and J. Stankovic, “Scheduling distributed real-time tasks with minimum jitter,” *IEEE Transactions on Computers*, vol. 49, no. 4, pp. 303–316, April 2000.
 - [71] L. David, F. Cottet, and N. Nissanke, “Jitter control in on-line scheduling of dependent real-time tasks,” in *Real-Time Systems Symposium, 2001. (RTSS 2001). Proceedings. 22nd IEEE*, London, England, December 2001, pp. 49–58.
 - [72] M. Nasri and M. Kargahi, “A method for improving delay-sensitive accuracy in real-time embedded systems,” in *Embedded and Real-Time Computing Systems and Applications (RTCSEA), 2012 IEEE 18th International Conference on*, Seoul, Korea, August 2012, pp. 378–387.
 - [73] K. Lin and A. Herkert, “Jitter control in time-triggered systems,” in *System Sciences, 1996., Proceedings of the Twenty-Ninth Hawaii International Conference on*, vol. 1, Maui, Hawaii, USA, January 1996, pp. 451–459.
 - [74] C.-C. Han, K.-J. Lin, and C.-J. Hou, “Distance-constrained scheduling and its applications to real-time systems,” *IEEE Transactions on Computers*, vol. 45, no. 7, pp. 814–826, July 1996.
 - [75] Z. Liu, H. Zhao, P. Li, and J. Wang, “An optimization model for IO jitter in device-level RTOS,” in *Information Technology: New Generations, 2009. ITNG ’09. Sixth International Conference on*, Las Vegas, Nevada, USA, April 2009, pp. 1528–1533.
 - [76] P. Pillai and K. Shin, “Real-time dynamic voltage scaling for low-power embedded operating systems,” in *Proceedings of the eighteenth ACM symposium on Operating systems principles*, Banff, Canada, October 2001, pp. 89–102.

- [77] T. Phatrapornnant and M. Pont, “Reducing jitter in embedded systems employing a time-triggered software architecture and dynamic voltage scaling,” *IEEE Transactions on Computers*, vol. 55, no. 2, pp. 113–124, 2006.
- [78] B. Mochocki, R. Racu, and R. Ernst, “Dynamic voltage scaling for the schedulability of jitter-constrained real-time embedded systems,” in *Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, San Jose, CA, USA, November 2005, pp. 446–449.
- [79] K. Richter, R. Racu, and R. Ernst, “Scheduling analysis integration for heterogeneous multiprocessor SoC,” in *Real-Time Systems Symposium, 2003. RTSS 2003. 24th IEEE*, Cancun, Mexico, December 2003, pp. 236–245.
- [80] W. Halang, “Achieving jitter-free and predictable real-time control by accurately timed computer peripherals,” *Control Engineering Practice*, vol. 1, no. 6, pp. 979–987, December 1993.
- [81] Atmel, *AT91SAM ARM-based Flash MCU*, February 2012.
- [82] S. Microelectronics, *STM32F405xx, STM32F407xx Datasheet*, May 2012.
- [83] M. Behnam and D. Isovici, “Real-time control and scheduling co-design for efficient jitter handling,” in *Embedded and Real-Time Computing Systems and Applications, 2007. RTCSA 2007. 13th IEEE International Conference on*, Daegu, Korea, August 2007, pp. 516–524.
- [84] A. Cervin, B. Lincoln, J. Eker, K. Årzén, and G. Buttazzo, “The jitter margin and its application in the design of Real-Time control systems,” in *Proceedings of the 10th International Conference on Real-Time and Embedded Computing Systems and Applications (RTCSA)*, Göteborg, Sweden, August 2004.

- [85] A. Cervin and J. Eker, “Control-scheduling codesign of real-time systems: The control server approach,” *Journal of Embedded Computing*, vol. 1, pp. 209–224, April 2005.
- [86] Y. Wu, G. Buttazzo, E. Bini, and A. Cervin, “Parameter selection for real-time controllers in resource-constrained systems,” *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 610–620, Nov. 2010.
- [87] A. Aminifar, P. Eles, Z. Peng, and A. Cervin, “Control-quality driven design of cyber-physical systems with robustness guarantees,” in *Design, Automation & Test in Europe (DATE 2013)*, Grenoble, France, March 2013, pp. 1093–1098.