

University of New Hampshire
University of New Hampshire Scholars' Repository

Master's Theses and Capstones

Student Scholarship

Spring 2009

Design and application of a wireless torque sensor for CNC milling

Jeffrey Scott Nichols

University of New Hampshire, Durham

Follow this and additional works at: <https://scholars.unh.edu/thesis>

Recommended Citation

Nichols, Jeffrey Scott, "Design and application of a wireless torque sensor for CNC milling" (2009). *Master's Theses and Capstones*. 452.
<https://scholars.unh.edu/thesis/452>

This Thesis is brought to you for free and open access by the Student Scholarship at University of New Hampshire Scholars' Repository. It has been accepted for inclusion in Master's Theses and Capstones by an authorized administrator of University of New Hampshire Scholars' Repository. For more information, please contact nicole.hentz@unh.edu.

DESIGN AND APPLICATION OF A WIRELESS
TORQUE SENSOR FOR CNC MILLING

BY

JEFFREY SCOTT NICHOLS
B.S., University of New Hampshire, 2007

THESIS

Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Master of Science
in
Electrical Engineering

May, 2009

UMI Number: 1466944

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

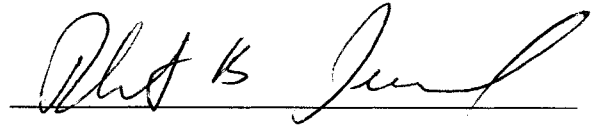
In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

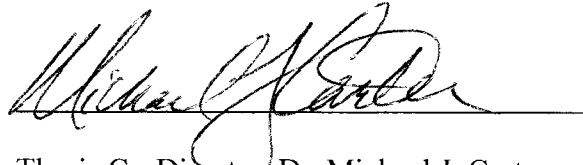
UMI Microform 1466944
Copyright 2009 by ProQuest LLC
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

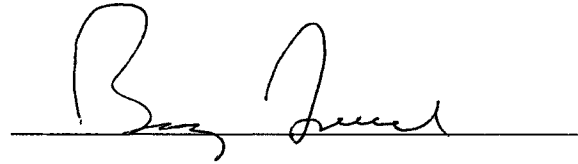
This thesis has been examined and approved.



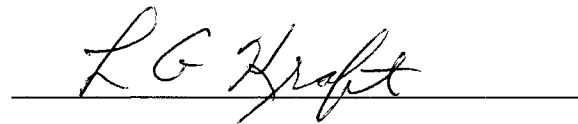
Thesis Director, Dr. Robert B. Jerard
Professor of Mechanical Engineering



Thesis Co-Director, Dr. Michael J. Carter
Associate Professor of Electrical and
Computer Engineering



Dr. Barry K. Fussell
Professor of Mechanical Engineering



Dr. L. Gordon Kraft
Professor of Electrical and Computer
Engineering



Dr. Kent A. Chamberlin
Professor of Electrical and Computer
Engineering

05-11-09

Date

DEDICATION

To my wife, Laura

ACKNOWLEDGMENTS

I would first like to thank Dr. Jerard for his enormous investment of time during the writing process of this thesis. His guidance and feedback was invaluable. Thanks to both Dr. Jerard and Dr. Fussell for their direction and insight during the course of my research. Thanks to Dr. Carter for his encouragement and help during my transition into graduate school. Thanks to Dr. Kraft and Dr. Chamberlin for finding time to be on my committee and review my thesis on short notice.

The support of the National Science Foundation under grant DMI-0620996 is gratefully acknowledged.

Special thanks goes to Chris Suprock, who developed the initial Smart Tool Holder. I am glad to have been able to contribute to the project.

I would like to thank my lab partners Brian, Chris, Corey, Cuneyt, Firat, Min, Raed, and Yanjun for their many insights and for providing a great working environment at the Design and Manufacturing Laboratory. I would also like to thank Adam Perkins, Kathy Reynolds, Tracey Harvey, and Sheldon Parent for the help they provided throughout my graduate studies.

Finally, I would like to thank my family. Without their love and support I would not be the person I am today.

TABLE OF CONTENTS

Dedication.....	iii
Acknowledgments.....	iv
List of Figures.....	viii
List of Tables.....	xi
Abstract.....	xii
Chapter 1: Introduction.....	1
1.1 Introduction.....	1
1.2 Thesis Overview.....	3
Chapter 2: Wireless Sensors.....	4
2.1 Introduction.....	4
2.2 Design Requirements.....	5
2.3 Design Description.....	9
2.3.1 Overview of the Smart Tool Holder Physical Design.....	9
2.3.2 Main Electronics Board Overview.....	11
2.3.3 Torque Measurement.....	13
2.3.4 Torque Drift Compensation.....	16
2.3.5 Temperature Measurement.....	17
2.3.6 Radio Transmitter.....	18
2.3.7 Processor.....	19

2.3.8 Indicator LEDs.....	19
2.3.9 Power.....	20
2.3.10 Battery Charging.....	24
2.3.11 Component Cost.....	25
2.4 Characterization and Calibration.....	25
2.4.1 Sampling Rate and Aliasing.....	25
2.4.2 Torque Calibration and Noise Level.....	27
2.4.3 Transmitter Latency.....	30
2.4.4 Battery Discharge Characteristics.....	31
2.4.5 Battery Charge Time.....	32
2.5 Summary.....	33
Chapter 3: Information Technology.....	34
3.1 Introduction.....	34
3.2 XML Schema.....	36
3.3 MATLAB Toolbox.....	39
3.4 Summary.....	42
Chapter 4: Applications of the Smart Tool Holder.....	43
4.1 Introduction.....	43
4.2 Force and Torque Model for a Chatter Simulator.....	46
4.2.1 Force Model.....	48
4.2.2 Torque Model.....	54
4.3 Chatter Simulator.....	55

4.4 Power Sensor.....	60
4.5 Summary.....	62
Chapter 5: Conclusions and Future Work.....	64
5.1 Conclusions.....	64
5.2 Future Work.....	65
5.2.1 Increase the Anti-Aliasing Filter Order.....	65
5.2.2 Replace Bluetooth Radio With Another Technology.....	65
5.2.3 Process Data On-Board.....	67
5.2.4 Add a Charging Temperature Sensor.....	67
5.2.5 Reduce Strain Gauge Temperature Sensitivity.....	68
5.2.6 Information Sharing Between Research Groups.....	69
5.2.7 eXist XML Database.....	69
5.2.8 Real-Time Controller for Chatter Suppression.....	71
5.2.9 Extension of Chatter Simulator.....	71
List of References.....	73
Appendix A: Smart Tool Holder Main Board Layout, Schematic and Parts.....	76
Appendix B: Charging Board Layout, Schematic and Parts.....	81
Appendix C: Additional Electronics Parts.....	85
Appendix D: XML Schema.....	86
Appendix E: Example XML Data File.....	89
Appendix F: Description of MATLAB Toolbox.....	92
Appendix G: Force and Torque Model Matrix Generation Program.....	97

LIST OF FIGURES

Figure 2.1: Smart Tool Holder (top) and sensor integrated cutting tool (bottom).....	9
Figure 2.2: Smart Tool Holder physical layout.....	10
Figure 2.3: Smart Tool Holder physical connection diagram.....	10
Figure 2.4: Smart Tool Holder main electronics board.....	12
Figure 2.5: Smart Tool Holder main electronics board block diagram.....	12
Figure 2.6: Strain gauge bridge balancing circuit.....	16
Figure 2.7: Battery voltage monitoring circuit.....	24
Figure 2.8: Torque calibration curve.....	28
Figure 2.9: Ambient noise and histogram.....	28
Figure 2.10: Noise power spectral density.....	29
Figure 2.11: Round-trip latency of 1000 samples.....	30
Figure 2.12: Battery discharge curve with typical load.....	32
Figure 3.1: XML Schema for experiment results file.....	36
Figure 3.2: Example usage of the Fourier Analysis Tool.....	39
Figure 3.3: Example usage of the Average Plotting Tool.....	40
Figure 3.4: Example usage of the Ktc, Kte Regression Tool.....	42
Figure 4.1: Block diagram of chattering milling system.....	44
Figure 4.2: Example stability lobe diagram [Suprock, et al. 2008].....	45
Figure 4.3: The angle of engagement.....	48

Figure 4.4: Angular limits of cutting.....49

Figure 4.5: Chip thickness due to dynamic displacement.....50

Figure 4.6: Tool runout magnitude and locating angle.....51

Figure 4.7: Differential forces on a slice of a cutting tool.....52

Figure 4.8: Simulink diagram for a mill.....55

Figure 4.9: Simulink diagram for cutting forces from displacement.....56

Figure 4.10: Simulink digram for displacement from cutting forces.....56

Figure 4.11: Basic usage for chatter simulator.....57

Figure 4.12: X and Y forces for a simulated cut at 4000 RPM (chatter).....58

Figure 4.13: X and Y forces for a simulated cut at 4300 RPM (stable).....58

Figure 4.14: X and Y forces for a simulated downmill using a 4-tooth cutter.....59

Figure 4.15: Power sensor data (filtered for clarity).....61

Figure 4.16: Smart Tool Holder data (filtered to show average torque).....62

Figure A.1: Top copper.....76

Figure A.2: Bottom copper.....76

Figure A.3: Top solder mask.....77

Figure A.4: Bottom solder mask.....77

Figure A.5: Top silkscreen.....77

Figure A.6: Bottom silkscreen.....77

Figure A.7: Main board electrical schematic.....78

Figure B.1: Top copper.....81

Figure B.2: Bottom copper.....81

Figure B.3: Top solder mask.....	82
Figure B.4: Bottom solder mask.....	82
Figure B.5: Top silkscreen.....	82
Figure B.6: Bottom silkscreen.....	82
Figure B.7: Charging board electrical schematic.....	83
Figure F.1: XML Measurement Toolbox accessed from the MATLAB Start menu.....	92
Figure F.2: Main GUI.....	93
Figure F.3: XML database selection dialog.....	93
Figure F.4: Load from files dialog.....	94
Figure F.5: Subset dialog box.....	95

LIST OF TABLES

Table 2.1: Power supply current requirements.....	22
Table 2.2: LDO efficiency and battery power consumption.....	23
Table 2.3: Battery charge times.....	33
Table A.1: Main electronics board components.....	79
Table B.1: Charging board components.....	84
Table C.1: Additional Smart Tool Holder parts.....	85

ABSTRACT

DESIGN AND APPLICATION OF A WIRELESS TORQUE SENSOR FOR CNC MILLING

by

Jeffrey Scott Nichols

University of New Hampshire, May, 2009

A Smart Machining System for Computer Numerical Control (CNC) Milling continually adjusts the cutting process parameters to optimize for cutting tool life and material removal rate. The system depends on sensors to gather information from the machine during cutting, but commercially available sensors detract from the effectiveness of the cutting system by lowering the system stiffness. This research focuses on the development of the electronics for a Smart Tool Holder (STH) and potential applications such as measurement of mechanical cutting power and suppression of chatter. The STH is a standard milling tool holder modified to hold a torque strain gauge bridge, a thermocouple and a Bluetooth radio transmitter. The STH is meant to overcome some of the different limitations imposed by bed dynamometers, microphones and spindle power sensors without reducing the system stiffness. Comparison of the mechanical power estimates from the STH and a conventional power sensor showed 10% difference.

CHAPTER 1

INTRODUCTION

1.1 Introduction

A Smart Machining System for CNC Milling [Xu 2007] utilizes sensors to gather information about the cutting process and responds by updating the feedrate and spindle speed of the cutting process to optimize various metrics, such as material removal rate (MRR), tool life or surface finish. Process updates can be made either off-line during the process planning stage or on-line during cutting.

A Smart Machining System is only as good as the information it gathers. There is a great need for sensors that can provide detailed information about the cutting process. It is desirable get this information without significantly altering the dynamics of the milling system. Historically, sensors used in machining have fallen into two categories: sensors that directly measure cutting forces and change the system dynamics, and sensors that don't alter dynamics and only indirectly provide information about the cutting forces.

An example of a sensor providing direct access to the cutting forces is a Kistler force dynamometer. It can provide cutting force signals with high accuracy, but it lowers the stiffness of the milling system. Lower stiffness adversely affects part tolerances and can cause chatter problems. Furthermore, the Kistler force dynamometer is an expensive piece of equipment and occupies space below the workpiece, reducing the machining

envelope. The high cost of the Kistler dynamometers (approximately \$30,000) and their invasive nature make them unsuitable for widespread adoption by manufacturing companies.

An example of a sensor that does not change dynamics is a microphone. The auditory data from a microphone can be used to measure relative tool wear by comparing the signal levels from a sharp tool to that of a worn tool [Desfosses 2007]. Microphone data can also be used to predict regenerative chatter through the use of systems such as the Harmonizer [Delio 1992]. Microphones, however, are prone to noise from external sources, such as other machines on a shop floor. Furthermore, microphones cannot easily be used to estimate the forces, making them unsuitable for some applications such as feedrate optimization [Jerard, et al. 2006].

Spindle motor power sensors do not alter the dynamics of the machining system, and can be utilized to estimate the mechanical cutting power. The mechanical power is directly proportional to the average tangential cutting force. From the mechanical power it is possible to perform feedrate optimization and tool wear monitoring [Desfosses 2007, Cui 2008]. To obtain the mechanical power, the electrical power of the unloaded spindle, or tare power, must first be measured. Because the tare power can change with spindle speed and motor temperature, acquiring a good estimate of the tare power can be difficult. Estimating the mechanical power also requires a motor efficiency curve, which is expensive and time-consuming to determine. Furthermore, the spindle power sensor has a time constant which is long relative to the tooth passing interval [Xu 2007]. Because of their long time constants, spindle power sensors are not suitable for

measuring high-frequency phenomena such as chatter.

Acquiring good information about the cutting process is only part of the problem, however. A Smart Machining System also needs to be able to communicate with its environment by providing sensor and status information for viewing and archival. There has been some work to standardize the inputs and outputs of machining systems through the use of open and extensible file formats. MTConnect [MTConnect] has been aimed at standardizing the status output across different component manufacturers, and NCML [Ryou 2001, Ryou and Jerard 2001, Schuyler 2005, Jerard and Ryou 2006] has been aimed at creating a standardized representation of the part to be cut. There is still, however, a need for a standardized file format for data archival for machining systems.

This research had two distinct focuses: the development and application of a wireless torque sensor, and the development of a new file format for archiving machining sensor data. These focuses, while distinct, are united in their goal of improved productivity for Smart Machining Systems and machine operators.

1.2 Thesis Overview

There are five chapters in this thesis. Chapter 1 provides background for the subject. Chapter 2 describes the requirements and development of a torque sensor for milling. Chapter 3 details a new file format designed to store cut configuration data alongside sensor data for the purposes of archival. Chapter 4 describes applications of the torque sensor. Chapter 5 gives a summary of the thesis and describes future work.

CHAPTER 2

WIRELESS SENSORS

2.1 Introduction

Sensors are an integral part of any information based machining framework. They are the eyes and ears of the system. Wireless sensors have the advantage of being capable of being deployed in locations where conventional sensors would be infeasible, such as on a rotating cutting tool. Sensors located on the cutting tool have more direct access to the cutting forces than to sensors located elsewhere in the machine, as the signal is not propagating through other material before being measured.

The cutting forces provide key information about the cutting process. They can be used to track wear, as the forces are lowest on a sharp tool, and increase over the lifetime of the cutting tool. Knowledge about the cutting forces can also be used during toolpath generation to optimize feedrates [Jerard, et al. 2006]. Furthermore, the cutting force frequency content can be analyzed to detect chatter modes as will be discussed in more detail in Chapter 4.

Wear monitoring and feedrate optimization have typically been performed using output from non-invasive spindle power sensors, but these power sensors do not have sufficient bandwidth to provide information about chatter conditions. Furthermore, it is necessary to know the spindle motor efficiency curve, requiring an expensive and time

consuming calibration. This efficiency curve can be difficult to determine outside of a research environment. Furthermore, the unloaded power of the motor, or tare power, changes with temperature, complicating the acquisition of accurate power measurements.

This chapter outlines the design requirements and development of the Smart Tool Holder, a milling tool holder with embedded torque and temperature sensors. Torque is directly proportional to the tangential cutting force. The Smart Tool Holder captures data from these sensors and transmits the signal wirelessly to a receiving computer where it can be recorded and analyzed. The purpose of the Smart Tool Holder is to provide the functionality of the power sensor for wear tracking and feedrate override, while also providing higher bandwidth information that can be used to analyze regenerative chatter.

Temperature plays a critical role in tool life [Drozda, et al. 1983]. Excessive heat accelerates tool wear. A temperature sensor embedded in the cutting tool may prove useful for applications in tool wear monitoring and tool wear research.

2.2 Design Requirements

The goal of the Smart Tool Holder is to capture high bandwidth torque data and low bandwidth temperature data during the milling process and send it wirelessly to a PC. The following list summarizes the design requirements for the Smart Tool Holder:

- The device must capture the DC component of torque
- The torque signal's bandwidth must be at least 2500Hz.
- The sampling rate on the torque signal must be high enough to prevent aliasing.
- Must have the RMS torque measurement noise below 0.15N-m.
- The device must be capable of measuring temperatures up to 900°C.

- The wireless transmitter must be capable of transmitting data without distortion.
- The wireless transmitter must be capable of sending data to a receiving PC at least 10 meters away in a machining environment.
- The device must have less than 100ms of latency between the acquisition of a torque sample and the reception of that sample on the PC.
- Provide a mechanism to bring the long-term drift in the torque signal to zero.
- Run for at least 1 hour after charging the device.
- Charge in less than two hours after an hour of usage.
- The physical design of the electronics must fit within the space between two concentric cylinders of height 5.08cm and diameters 4.45cm and 5.71cm.
- The cost of the electronics should be less than \$150.

Each of these requirements is meant to ensure that the device is usable in a machining environment and useful for machining related research.

The DC component of torque is important for capturing the average mechanical power. The average power can be used to estimate the coefficients for the cutting model described in [Altintas 2000], which can be used as a cutting-condition independent indicator of tool wear [Desfosses 2007, Cui 2008]. The mechanical power is a simple function of the torque T and the spindle speed:

$$P(t) = T(t) \omega = T(t) \frac{2\pi \text{RPM}}{60}, \quad (2.1)$$

where T is given in N-m, and P is in W. Therefore, to calculate the average power, the average torque (or DC torque) is needed.

The 2500Hz requirement on bandwidth is meant to ensure that the device can capture the frequencies of interest with minimal distortion. Implicit in this requirement is that the passband be relatively flat. A bandwidth of 2500Hz allows the device to capture the tooth passing frequency and nineteen harmonics for a two tooth cutter rotating at 7,500 RPM.

The sampling rate of the torque acquisition system must be high enough to prevent high frequency components from aliasing into the signal's passband. The Nyquist-Shannon sampling theorem states that the sampling frequency must be at least twice the bandwidth of the signal, though in practice the sampling rate must be much higher.

The RMS level of the noise must be below 0.15N-m. This is to allow torques from a cut with low forces to be measured. The maximum measurable temperature must be larger than 900°C to ensure that the device can measure typical cutting temperature [Drozda, et al. 1983].

The requirement for no digital distortion is based on fact that the analysis algorithms have not yet been firmly established, and lossy compression may render certain types of analysis ineffective. Once the analysis algorithms are more firmly established and understood, some types of lossy compression may become acceptable.

The wireless transmitter must be capable of sending data over a distance of at least 10 meters in a machining environment. This is based on the expected use of the device. It will be used in noisy machining environments (e.g. spindle motor running, coolant running, and metallic chips being removed from the workpiece), so the

transmitter must be capable of handling the interference without degrading the performance. The 10 meter requirement is meant to ensure that the device can be effectively used with any placement of the receiving PC.

The latency requirement of less than 100ms between the sampling of torque and the reception of the sample on the PC is meant to provide a upper bound on the delay in the feedback path for the analysis of control strategies. This is primarily aimed toward chatter avoidance control strategies, since the detectable buildup of regenerative chatter is on the order of seconds. This is in contrast to tool breakage, which can be on the order of milliseconds. Tool wear, on the other extreme, happens over minutes or hours.

Since many torque sensor technologies drift over their lifetime, the device should contain a mechanism to compensate for this drift, either in software or in hardware. Without compensation, the DC component of torque would only be valid directly after a calibration, and power estimation would be prone to error.

The hour run time requirement is there to allow the device to perform over the course of a typical cutting run. Implicit in this requirement is that the performance of the device be consistent over the hour of running. A shorter run time would limit the general usefulness of the device.

Similarly, the charge time requirement is meant to limit the downtime of the tool. With too long a charging period, the tool could not be used often enough to make it a useful product.

The constraint on the physical dimension of the electronics is imposed by the rest of the design for the Smart Tool Holder. The stock tool holder has a diameter of 4.45cm,

and the protective shroud has an inner diameter of 5.71cm, and the electronics must fit within the space between these. The height of 5.08cm is imposed by the height of the base tool holder and the location of its set screw.

The cost of the components must be less than \$150 to ensure that the finished product is economical to produce.

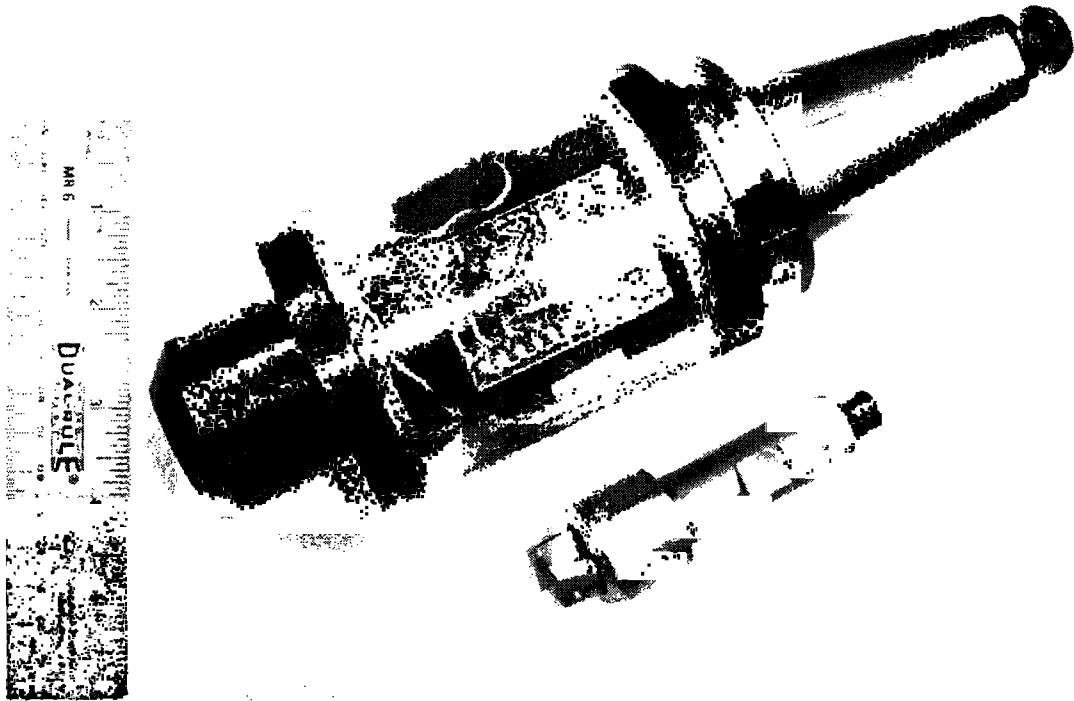


Figure 2.1: Smart Tool Holder (top) and sensor integrated cutting tool (bottom)

2.3 Design Description

2.3.1 Overview of the Smart Tool Holder Physical Design

The physical realization of the Smart Tool Holder is shown in Figure 2.1. The device consists of a tool holder and a removable cutting tool. The tool holder was created from a commercially available Parlec Weldon Shank C40 tool holder with a 6" (15.24cm) overhang, modified to house the main electronics (detailed in Appendix A), battery and charging circuitry (detailed in Appendix B) in a sealed shell. The removable cutting tool

is a Sandvik 390 insert holder that has been modified to hold the torque and temperature sensors. The locations of the major components are indicated in Figure 2.2 and the connections between them are shown in block diagram form in Figure 2.3.

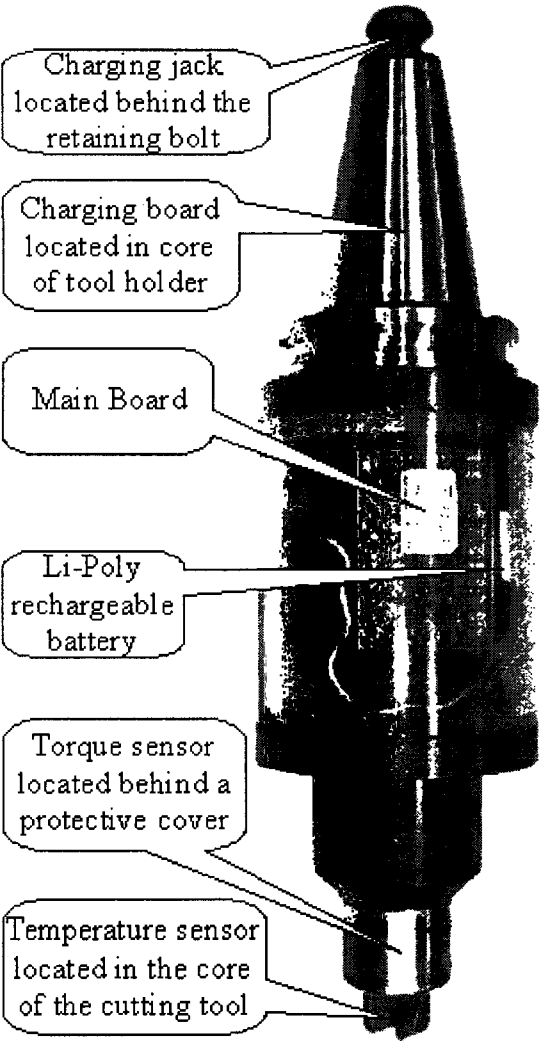


Figure 2.2: Smart Tool Holder physical layout

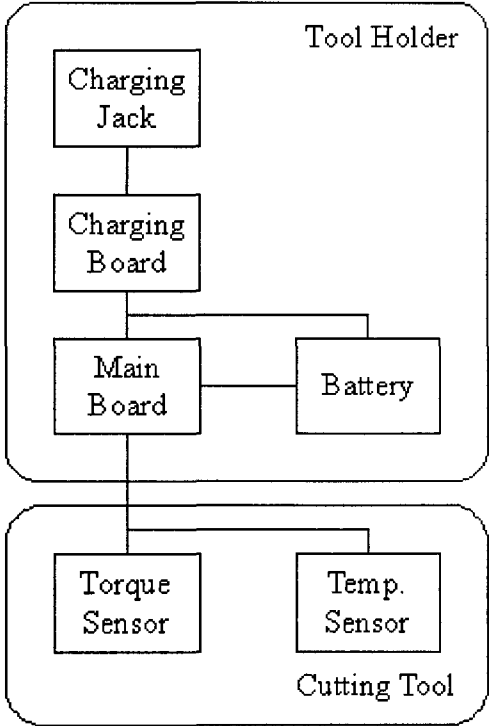


Figure 2.3: Smart Tool Holder physical connection diagram

The charging jack is located behind the spindle retaining bolt to protect it from the fluid and metal chips present during cutting operations. The charging board handles all of the battery monitoring and voltage regulation tasks necessary during the charging of the

Lithium-Polymer battery. The main electronics and battery are both housed within a transparent Lexan shroud around the tool holder body to allow viewing of the indicator lights while still providing sufficient protection to the components during the metal cutting process.

The torque strain gauge bridge is mounted on the shank of the removable cutting tool and covered by a protective aluminum shield. This shield prevents the strain gauges from being damaged by sharp chips being produced at the cutting interface. The thermocouple for measuring the temperature is located along the central axis of the removable cutting tool, equidistant between the two cutting inserts. A connector on the back of the tool mates with a connector in the core of the tool holder and provides an electrical connection between the sensors and the main electronics.

2.3.2 Main Electronics Board Overview

The complete layout, electrical schematic and parts list for the main electronics board is contained in Appendix A. The main electronics board (see Figure 2.4) holds the processor, radio transmitter and circuitry necessary for amplifying and digitizing the sensor signals. The crystal oscillator helps the processor maintain the accurate timing necessary for communicating with the radio module. The torque signal is passed through an instrumentation amplifier to increase the signal magnitude and then it is put into an analog to digital converter (A/D converter). A digital to analog converter (DAC) adds or removes current from one side of the torque strain gauge bridge to bring the output to zero. The thermocouple signal is put into a MAX6675 chip, a specialized temperature to digital converter that performs all the necessary steps to convert the non-linear

thermocouple output into an accurate temperature measurement. An interconnect diagram for these electrical systems on the main board is shown in Figure 2.5. The dimensions of the board are 3.048cm in width and 4.826cm in height. These dimensions ensure that it can fit between the tool holder body and the Lexan shroud.

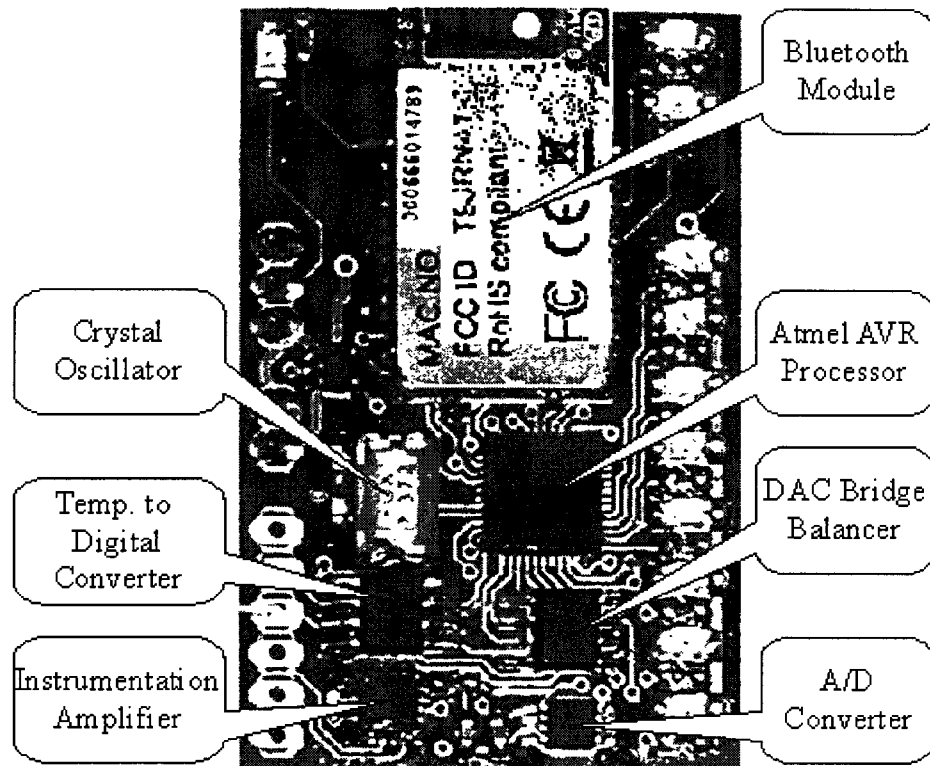


Figure 2.4: Smart Tool Holder main electronics board

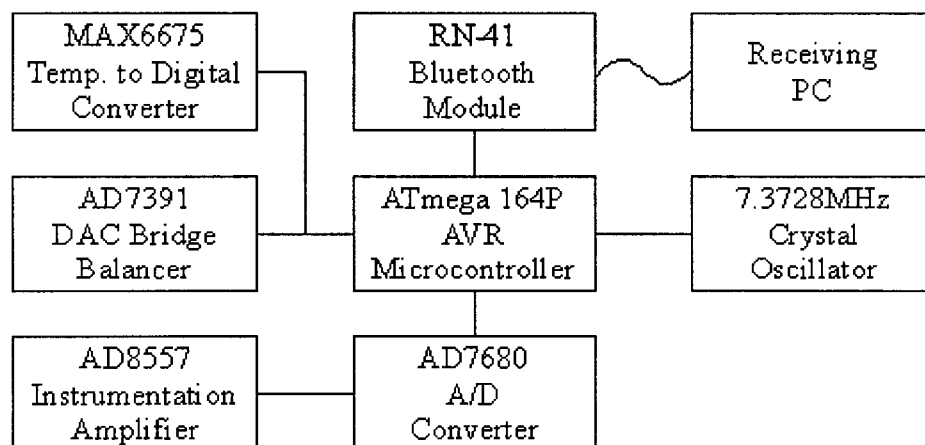


Figure 2.5: Smart Tool Holder main electronics board block diagram

2.3.3 Torque Measurement

Strain gauges configured in a Wheatstone bridge can measure the torsional strain on a shaft, while rejecting the strain from bending and compression. The bridge also helps to reduce the effects of thermal strain. The torsional strain, τ , that the bridge measures on the removable cutting tool is given by:

$$\tau = \frac{T D}{2 J G}, \quad (2.2)$$

where T is the torque in N-m, D is the diameter in meters, J is the polar moment of inertia in m^4 , and G is the shear modulus of the material in Pascals. The polar moment of inertia, J , of the cutting tool can be approximated by the moment of inertia for a solid cylinder:

$$J = \frac{\pi}{32} D^4. \quad (2.3)$$

Using a diameter of 1.905cm and a typical shear modulus for steel of 80GPa, the relationship between torsional strain and torque on the removable cutting tool becomes:

$$\tau = T \frac{3.885 * 10^{-6}}{N \cdot m}. \quad (2.4)$$

The change in voltage, ΔV , across the strain gauge bridge is a function of the gauge factor of the strain gauges, k_s , the torsional strain, τ , and the applied voltage, V :

$$\Delta V = k_s \tau V. \quad (2.5)$$

Using an applied voltage of 3.1V (the rationale behind this voltage is discussed in Section 2.3.9), the ratio between torque and output voltage is:

$$\frac{\Delta V}{T} = k_s \frac{12.04 * 10^{-6} V}{N \cdot m}. \quad (2.6)$$

This relationship is only an estimate, since the exact shear modulus of the material was not measured. The equation does, however, convey order of magnitude information about the relationship.

Wire strain gauges were chosen because they have a gauge factor of $k_s=2$, which is sufficient to get the minimum torque of $0.15 \text{ N}\cdot\text{m}$ into the microvolt range. This gives a theoretical voltage to torque ratio of $24.08 * 10^{-6} * \text{V}/\text{N}\cdot\text{m}$. Semiconductor strain gauges provide a much higher gauge factor, but at a higher cost. Semiconductor gauge factors are typically anywhere from 100 to 150.

Because the output voltage of the bridge is small, an instrumentation amplifier is needed to amplify the signal into a voltage range usable by the A/D converter. The instrumentation amplifier also converts the differential signal into a single ended signal for use with a standard A/D converter. The Analog Devices AD8557 instrumentation amplifier was selected because it was one of only a few devices that provided programmable gain. The programmable gain allows the microcontroller to reduce the scale of the torque signal during heavy cuts, preventing the A/D converter from saturating and clipping the waveforms. The alternative is to select an instrumentation amplifier with a fixed gain low enough to encompass the heaviest cutting conditions, but the drawback is decreased sensitivity to lighter cutting conditions. The AD8557 has a minimum gain of 28, and a maximum gain of 1300. The gain of 1300 is the default gain in this design.

The A/D converter selected was the Analog Devices AD7680 as it contained a Serial Peripheral Interface (SPI), had 16-bit resolution, and provided sufficient sampling rate in a small package with a low cost. The SPI port allows simple interfacing with a

microcontroller. Many other A/D converters would have provided similar functionality.

A first order RC low-pass filter between the instrumentation amplifier and the A/D converter acts as an anti-aliasing filter. A 5.6kohm resistor and 10nF capacitor provide a corner frequency of approximately 2.84kHz. This provides more than the required 2.5kHz bandwidth. The sampling frequency of the A/D converter is set at 10.24kHz. The testing for aliasing is described in Section 2.4.1.

The gain of 1300 in conjunction with a 16-bit A/D means that the resolvable voltage on the bridge is:

$$\Delta V = \frac{3.1V}{1300 * 2^{16}} = 39.39 \text{ nV} . \quad (2.7)$$

The resolvable torque is (assuming a shear modulus of 80GPa):

$$T = \frac{\Delta V}{12.04 * 10^{-6} \text{ V/N}\cdot\text{m}} = 0.003021 \text{ N}\cdot\text{m} . \quad (2.8)$$

This resolution, however, is only possible in a system without noise (or in a system with almost no bandwidth). Since the strain gauges are resistive, they are subject to Johnson-Nyquist noise, or thermal noise. The RMS voltage for this noise is given by:

$$V_{noise} = \sqrt{4 k_B T R \Delta f} , \quad (2.9)$$

where k_B is Boltzmann's constant, T is the Kelvin temperature, R is the resistance of the bridge, and Δf is the width of the frequency range being sampled. A first order filter, such as the one employed as an anti-aliasing filter, has a noise equivalent bandwidth of 1.5 times the corner frequency. For the 350 ohm strain gauge bridge at room temperature, the thermal noise is:

$$V_{bridge} = \sqrt{4 \cdot 1.38066 \cdot 10^{-23} \text{ J/K} \cdot 300 \text{ K} \cdot 350 \Omega \cdot 1.5 \cdot 2.84 \text{ kHz}} = 157.17 \text{ nV} . \quad (2.10)$$

Another source of noise is the instrumentation amplifier. The AD8557 is rated at an input noise level of 32 nV/ $\sqrt{\text{Hz}}$. The voltage noise from this is:

$$V_{amp} = 32 \text{ nV}/\sqrt{\text{Hz}} \cdot \sqrt{1.5 \cdot 2.84 \text{ kHz}} = 2.089 \text{ uV} . \quad (2.11)$$

This is a significantly larger source of noise than the strain gauges. The combined noise from these sources is:

$$V_{noise} = \sqrt{(157.17 \text{ nV})^2 + (2.089 \text{ uV})^2} = 2.095 \text{ uV} . \quad (2.12)$$

The torque corresponding to the voltage noise is:

$$T_{noise} = 2.095 \text{ uV} / (24.08 \text{ uV/N} \cdot \text{m}) = 0.0870 \text{ N} \cdot \text{m} . \quad (2.13)$$

This means that the minimum torque of 0.15 N·m is, at least in theory, feasible. The testing to verify this result is described in Section 2.4.2.

2.3.4 Torque Drift Compensation

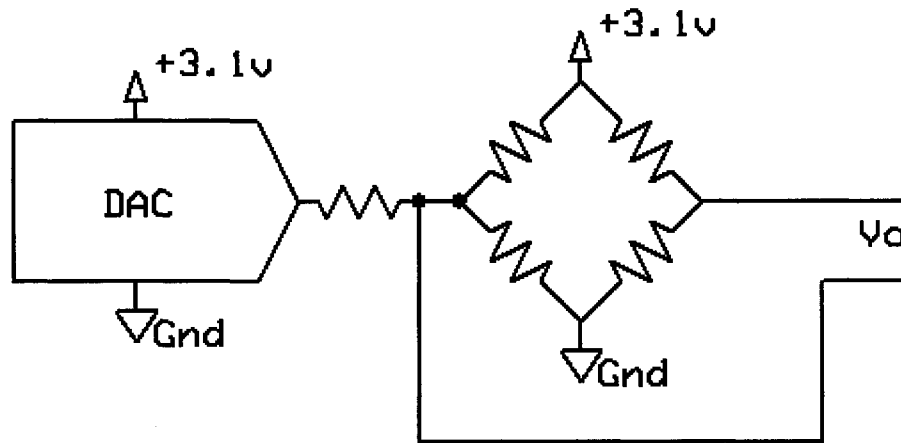


Figure 2.6: Strain gauge bridge balancing circuit

Manufacturing variation of the individual strain gauges and drift over the product lifetime can cause the output of a Wheatstone bridge to have a measurable offset at zero

torque. To compensate for this, a DAC was added to apply voltage through a resistor to one side of the bridge in order to bring the differential output signal to zero (see Figure 2.6). This process of balancing the bridge is performed whenever the Smart Tool Holder is powered on (this adds an additional requirement that there be no applied torque when the device is activated).

The DAC selected was a AD7390. It was selected because it provided an SPI interface (again, for ease of connecting to the microprocessor), had sufficient accuracy for balancing the strain gauge bridge, and had a low cost. Again, the specific choice of the DAC was more or less arbitrary, and many other converters would have provided similar functionality.

2.3.5 Temperature Measurement

Thermocouples were chosen for the temperature sensing technology because of their wide temperature range. Type-K thermocouples were selected because of their availability and the existence of commercial compensation chips. The MAX6675 was selected for the cold junction compensation chip because it handles the digitization directly. Cold junction compensation removes the dependence on the ambient temperature from the measurement of the thermocouple. This allows simple and easy integration with the microcontroller.

Resistive temperature devices (RTDs) were rejected as a possibility because their low tolerance for vibration and because of their smaller input range. The alternative to the MAX6675 was to use an instrumentation amplifier and A/D to determine the voltage on the thermocouple and an additional temperature sensing IC to determine the cold junction

temperature. These two measurements could then be transformed into a temperature measurement. This possibility was rejected because of the additional complexity and board space usage, though it does have the potential to offer more accuracy than the MAX6675.

2.3.6 Radio Transmitter

The Roving Networks RN-41 Bluetooth Module was selected for the role of transmitting data to a PC. The use of Bluetooth [Bluetooth] allows any PC with Bluetooth support to utilize the device. Additionally, for computers without native Bluetooth support, there are commercially available dongles that add Bluetooth functionality to USB enabled PCs. The RN-41 uses the Bluetooth Serial Port Profile (SPP) and appears as a virtual COM port on the host computer, providing a simple interface for the software developer.

While the Bluetooth audio profiles could conceivably be used for sending the sensor data, they would introduce undesirable distortion as part of the compression, and may lose portions of the data due to interference. The SPP on the other hand does not alter the data, and provides a reliable connection (meaning data not received is retransmitted).

Many of the Bluetooth alternatives do not have sufficient transmission rates to stream 160kbps (16-bit samples at 10kHz) to a PC. Those that do have acceptable transmission rates require extra design work to develop the receiving hardware for the PC side. For this reason, Bluetooth was selected for the radio protocol. The RN-41 was chosen over other Bluetooth modules because of its compact size and ease of use. The

RN-41 is advertised as supporting sustained data rates up to 240kbps, and a range of up to 100 meters.

Latency is not specified for the RN-41, so testing was required. This procedure is described in Section 2.4.3.

2.3.7 Processor

The Atmel AVR series of 8-bit microcontrollers was selected for the central processor. Specifically, the ATmega164P was chosen because it offers the ability to have two separate SPI ports in addition to a universal asynchronous receiver transmitter (UART). The two SPI ports are necessary because the A/D converter's SPI connection transmits much more quickly than the DAC's and thermocouple converter's SPI connection can handle. Having separate SPI ports for the high and low speed transmissions simplifies the design of the hardware and software. The UART is needed to communicate with the RN-41 Bluetooth Module. Because of the high degree of timing accuracy needed for reliable UART transmission, an external crystal oscillator was provided to the microcontroller. The oscillator frequency of 7.3728 MHz allows integer division to the UART baud rates supported by the RN-41.

Many other microcontrollers would be suited for the task of processing on the Smart Tool Holder, but the AVR series of microcontrollers was selected because an Atmel STK500 AVR hardware programmer was already available during the development phase of the electronics.

2.3.8 Indicator LEDs

There are eight LEDs attached to the microcontroller that can give feedback about

the current status of the Smart Tool Holder to a user. These lights are each independently controllable by the microcontroller. There are three green lights, three yellow lights and two red lights in a vertical line along the right edge of the main electronics board.

The RN-41 Bluetooth module is connected to two indicator LEDs. The first, a blue LED, indicates the current device state by different intervals of blinking. It blinks slowly when the RN-41 is in discoverable mode, blinks quickly when in configuration mode, and is solid during an active connection. The second LED, a green one, only shows connection status. It turns on when the device has an established connection to a PC, and is off otherwise. Further details about these indicators can be found in the RN-41 manual.

There is also a red LED that indicates battery charging status. The charging circuitry turns the indicator on when the battery is being charged, and off otherwise.

2.3.9 Power

Because the Smart Tool Holder's electronics use a significant amount of power (see Table 2.1), only battery based power systems were considered. Solar panels and energy harvesting devices weren't capable of supplying the necessary power without introducing extra complications or prohibitive expenses. Lithium-Polymer was selected for the battery technology because of its high energy density, allowing small, light-weight batteries to power the device.

A 430mAh capacity battery was selected to power the electronics, which provides a working time of around 4 hours between charges under normal usage. Larger capacity batteries did not meet the space requirements imposed by the shroud. This battery has a maximum discharge rate of 6A and a maximum charge rate of 0.4A. The design only

calls for one battery, but with careful cable management, there is sufficient space within the shroud to add a second battery in parallel with the first. A second battery would effectively double the running time of the Smart Tool Holder, but would also double the charging time (without redesigning the charging circuitry to safely support charging two batteries at maximum current). Further discussion will be limited to Smart Tool Holders containing only a single battery.

Since the battery voltage varies with the remaining capacity, it is necessary to use a voltage regulator to provide a constant supply voltage. A single 3.1V low-dropout (LDO) regulator was chosen to power the strain gauge bridge, conditioning circuitry and the digital electronics. The LED indicator lights use the unregulated supply since variations in output intensity are acceptable. Another alternative to the LDO regulator was a buck converter, a switching power supply designed to output a lower voltage than the input voltage. The buck converter has higher power efficiencies when the input voltage is significantly higher than the output voltage, but has similar efficiencies to a LDO for the case of a 3.7V Lithium-Polymer battery with 3.1V output voltage. Buck converters were rejected because their higher component cost, increased power supply noise, and larger physical footprint.

The acceptable regulated voltage was between 3.0V and 3.6V as determined by the minimum and maximum supply voltage rating of the various components. The lower the chosen supply voltage, the more battery power can be extracted before dropping below the acceptable level. For this reason, the 3.1V level was chosen, to allow variation in the regulator to stay within the component limits, while still taking advantage of most

of the battery capacity.

Component	Max. Current (mA)	Average Current (mA)
RN-41 Bluetooth Module	100	30
ATMega164P Microcontroller	9	8
MAX6675 Temp. to Digital Converter	1.5	0.7
AD7680 A/D converter	2.8	2.8
AD7390 DAC	0.1	0.1
AD8557 Instrumentation Amplifier	1.8	1.8
350 ohm Strain Gauge Bridge	8.9	8.9
Total (Regulated)	124.1	52.3
LED Indicator Lights (Unregulated)	100	50
Total (Unregulated)	224.1	102.3

Table 2.1: Power supply current requirements

The maximum and average current usages for the various components are shown in Table 2.1. The average current listed for the RN-41 Bluetooth Module is the average during transmission. The current usage is around 25mA when not connected to a PC. From the table, the sum of the regulated maximum currents is 124.1mA, meaning that the regulator must be able to supply a peak demand of at least this much.

LDO regulators have power efficiencies of approximately V_{OUT}/V_{IN} , since the current into the regulator, I_{IN} , is nearly the same as the current out of the regulator, I_{OUT} (there is a small quiescent current, I_q , but it is generally much, much smaller than I_{OUT}):

$$\eta_{LDO} = \frac{P_{OUT}}{P_{IN}} = \frac{V_{OUT} * I_{OUT}}{V_{IN} * I_{IN}} = \frac{V_{OUT} * I_{OUT}}{V_{IN} * (I_{OUT} + I_q)} \approx \frac{V_{OUT}}{V_{IN}} \quad (2.14)$$

So for a 3.1V regulator running off a Lithium-Polymer battery with a maximum cell voltage of about 4.0V, the efficiency is 77.5%. Since the input voltage drops as the battery is depleted, the efficiency increases to a maximum of 96.9% just before the output

voltage starts to drop below its regulation level (assuming a regulator drop out of 100mV, where $V_{IN} = 100\text{mV} + 3.1\text{V}$). The efficiencies and power consumptions are summarized in Table 2.2.

Battery Voltage	LDO Efficiency (%)	Max. Power (mW)	Average Power (mW)
Full (4.0V)	77.5	896	409
Typical (3.7V)	83.8	829	379
Empty (3.2V)	96.9	717	327

Table 2.2: LDO efficiency and battery power consumption

Ideally, the 430mAh capacity battery will supply the average current of 102.3mA for 252 minutes. Similarly, the battery will supply the maximum current of 224.1mA for 115 minutes. In practice, however, the voltage from the battery will drop below the acceptable V_{IN} of 3.2V before the battery is completely depleted. The exact amount of usable capacity depends on the discharge rate, battery age, and temperature, but is generally between 80% and 95%. Even so, the worst case scenario of continuous 224.1mA of current with only 52% of the capacity being usable still meets the design requirement of a 1 hour run time. The run-time testing results are shown in Section 2.4.4.

The Texas Instruments TPS73131 regulator was chosen for this design. It allows input voltages up to 5.5V and has a drop out of 100mV. It can supply a constant 150mA, above the necessary peak demand of 124.1mA. Many similar regulators exist, but the TPS73131 was chosen because of its low cost.

One additional design feature was to include two 100kohm resistors in a voltage divider configuration across the battery, with the center terminal going to one of the processor's internal A/D converters (the internal A/D converters were not used for torque because of insufficient resolution and sampling rate). Figure 2.7 shows a schematic of

this monitoring circuit (the schematic of the entire board is contained in Appendix A). Using this circuit, the processor can monitor a battery voltage up to 6.2V (voltages beyond this level violate the microcontroller's input voltage ratings).

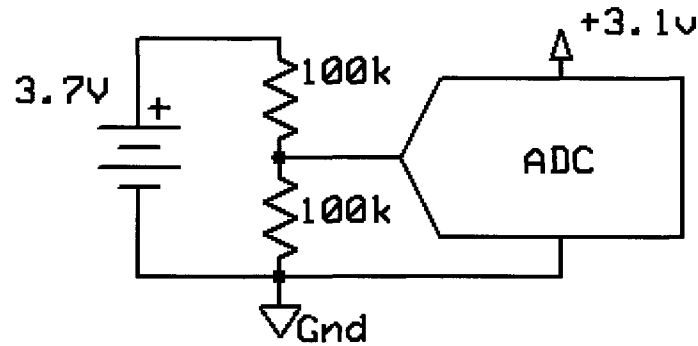


Figure 2.7: Battery voltage monitoring circuit

2.3.10 Battery Charging

The complete layout, schematic and parts list for the charging board is contained in Appendix B. Because of the complexities involved with charging Lithium-Polymer batteries, a commercial battery charging chip was selected. The TI BQ2057 advanced linear charge management IC provides conditioning, constant current, and constant voltage charging. The BQ2057 was selected for its simple integration and low cost. Another charging solution, the Texas Instruments BQ24001 was another option with the advantage of not needing an external transistor, but it was eventually rejected because of its requirement of soldering a thermal pad to the PCB. This requirement could not be met by the production capabilities during prototyping.

The charging board with the BQ2057 was designed to limit the charging current to 350mA. This limit was to keep the charging current below the battery's 400mA limit. The exact charging time depends on how deeply the battery was depleted. The charge time

testing is described in Section 2.4.5.

2.3.11 Component Cost

The cost of the individual main electronics board components is listed in Appendix A. These parts total to approximately \$82.92. The cost of the charging board components is listed in Appendix B. These parts total to approximately \$4.72. The miscellaneous other electronic components necessary for the design are listed in Appendix C. These parts total to approximately \$33.14. The sum of these component costs is \$120.78. This does not include the printed circuit board fabrication cost, which is roughly \$20 at low volumes per pair of main board and charging board. This brings the cost of materials to around \$140, which is less than the target of \$150.

2.4 Characterization and Calibration

While most of the design requirements were met by component choice, some needed to be tested to ensure compliance. The list of design requirements not guaranteed by component choice is:

- The sampling rate on the torque signal must be high enough to prevent aliasing.
- Must have the RMS torque measurement noise below 0.15N-m.
- The device must have less than 100ms of latency between the acquisition of a torque sample and the reception of that sample on the PC.
- Run for at least 1 hour after charging the device.
- Charge in less than two hours after an hour of usage.

2.4.1 Sampling Rate and Aliasing

The corner frequency of the first-order, low-pass, anti-alias filter is 2.84kHz. The

magnitude of the transfer function for the filter is given as follows:

$$|H(f)| = \frac{1}{\sqrt{1+(f/2.84\text{kHz})^2}} \quad (2.15)$$

This means that at the corner frequency of 2.84kHz, the torque signal will be attenuated to 70.71% of its original value. The filter is designed to prevent high frequency components from entering the A/D converter where they can alias into parts of the spectrum containing the torque information. The sampling frequency of the A/D converter is 10.24kHz. The aliasing isn't an issue until the frequencies start aliasing back into the passband. This happens when the frequencies pass above $10.24\text{kHz} - 2.84\text{kHz} = 7.40\text{kHz}$. At this frequency, the signal magnitude is 35.85% as calculated by equation (2.15). Frequencies at the sampling rate get aliased to DC, and their magnitude is 26.74%.

These attenuation levels are not sufficient to prevent noticeable aliasing. There are several solutions to this. The most obvious solution is to increase the order of the filter to provide more attenuation at the higher frequencies. The current hardware cannot, however, be easily modified to support a higher filter order, so a redesign would be necessary. Another solution is to increase the sampling rate of the A/D converter. This gives a higher Nyquist frequency, giving the existing filter more opportunity to filter frequencies before they alias. A significantly higher sampling rate cannot be easily achieved by the current system for two reasons. The first reason is that the Bluetooth transmitter is limited to 260kpbs, meaning the maximum possible theoretical sampling rate is 16.25kHz, though in practice the limit is closer to 12kHz due to protocol overhead. The second limitation is that a large portion of the current system's processing power is

already consumed by handling the 10.24kHz sampling rate, and a higher rate may exceed the capabilities of the microcontroller. Another solution is to reduce the bandwidth of the device by lowering the anti-alias filter's corner frequency. To be effective, however, lowering the corner frequency would require the bandwidth to be reduced below the level of the design requirement of 2.5kHz.

The current configuration of the system does not sufficiently attenuate high frequency signals, but on the other hand, the mechanical systems which the Smart Tool Holder is designed to monitor do not typically generate high frequency signals. The high frequency components being measured are primarily noise. This means that the device will experience more noise than otherwise expected.

2.4.2 Torque Calibration and Noise Level

The torque was calibrated using weights hung from a moment arm. The tool holder was mounted horizontally in a vice. A long clamp was attached to the insert holder to provide a place to hang weights to generate a torque on the instrument. The setup was leveled, and five weights of varying sizes were hung one at a time from the clamp. The results of the calibration are shown in Figure 2.8. The length of the moment arm was 27.07cm. The weights had masses of 100g, 200g, 500g, 1kg, and 2kg.

The voltage to torque ratio of $34.64 * 10^{-6} * \text{V}/\text{N}\cdot\text{m}$ was found using linear regression on the five data points. This is close to the theoretical voltage to torque ratio of $24.08 * 10^{-6} * \text{V}/\text{N}\cdot\text{m}$. This difference can be explained by either a lower shear modulus than 80GPa or a lower polar moment of inertia due to the complex geometries near the strain gauge bridge.

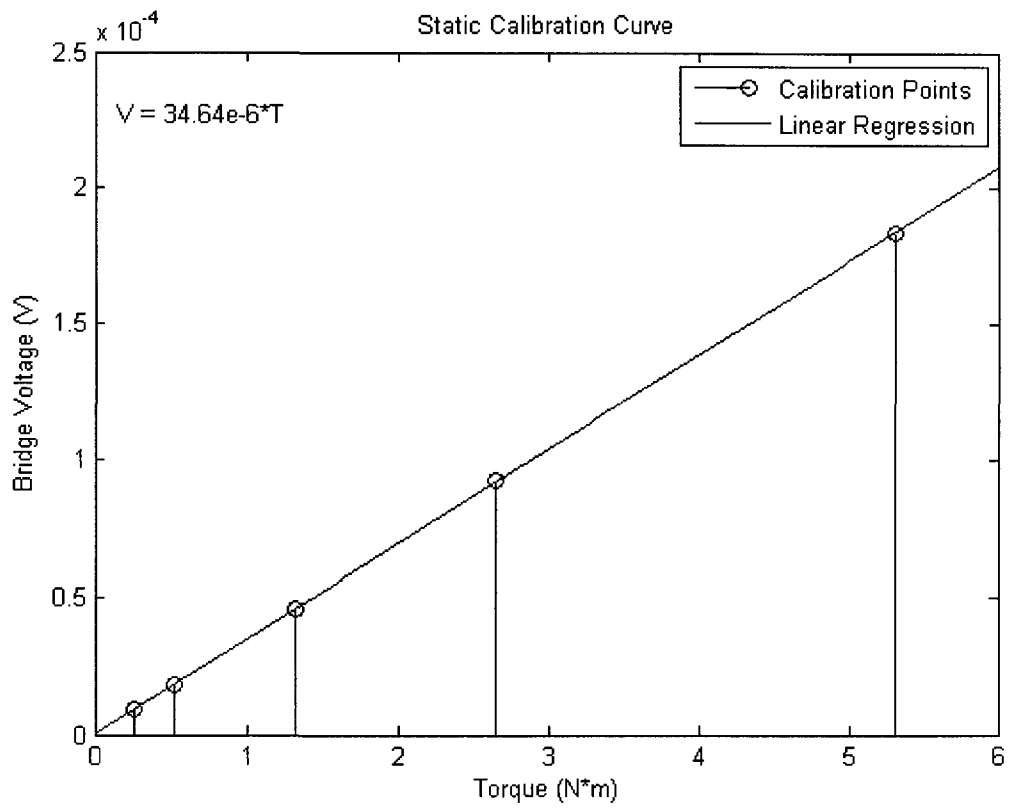


Figure 2.8: Torque calibration curve

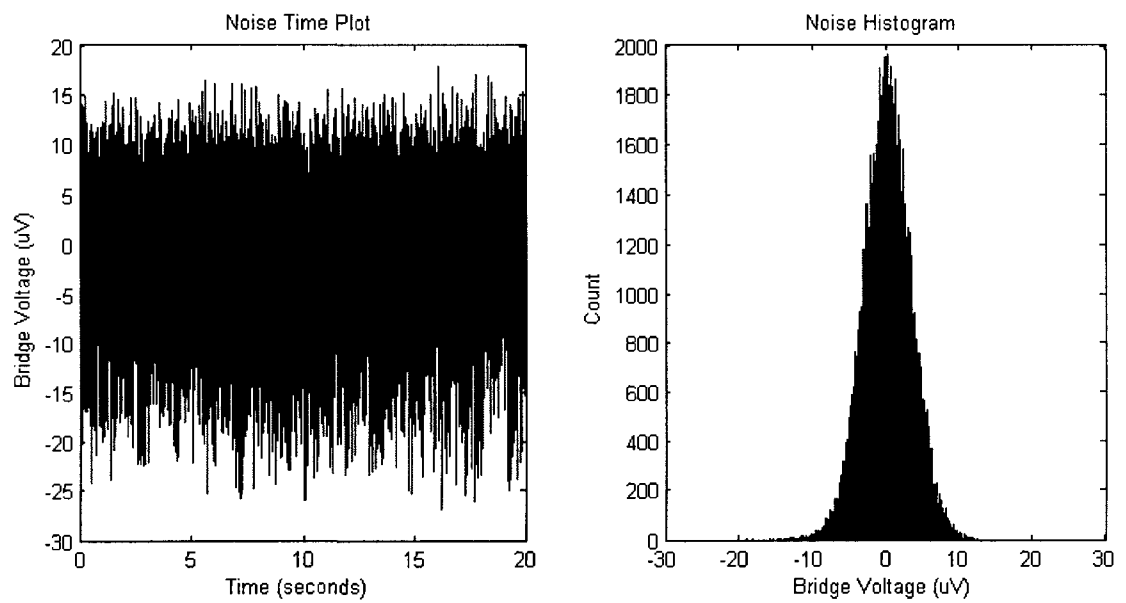


Figure 2.9: Ambient noise and histogram

The ambient noise level with no torque applied was recorded over a period of 20 seconds. Both a time plot and a histogram of the noise are shown in Figure 2.9. Additionally, a Power Spectral Density (PSD) estimate of the noise is shown in Figure 2.10. The PSD estimate is computed using Welch's method of averaging windowed periodograms. There are several frequency components in the spectrum that are not accounted for by white Gaussian noise. The frequency components at 800Hz and 1600Hz may be coupled from the RN-41 as these match the Bluetooth frequency hopping intervals. The large spike at low frequencies is 60Hz noise. The other frequency components are likely coupling from the other electronics, but the exact source is unknown.

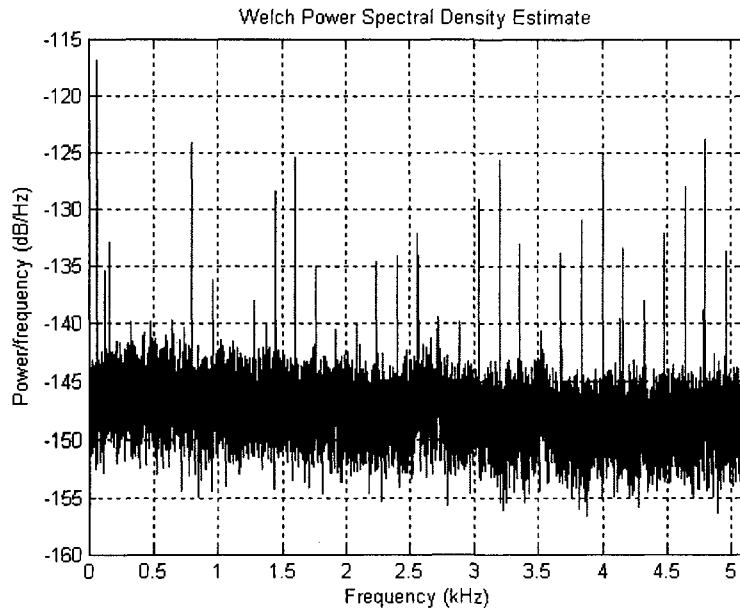


Figure 2.10: Noise power spectral density

The RMS level of the noise is 3.498uV. This level is higher than the 2.095uV expected by theory. Part of this noise may be due to high frequency noise aliasing to lower frequencies, as discussed in Section 2.4.1. Converting this voltage into a torque

gives $0.1010 \text{ N}\cdot\text{m}$. This level meets the design requirement of less than $0.15 \text{ N}\cdot\text{m}$ of noise.

2.4.3 Transmitter Latency

The latency between the capturing of a piece of data and its reception by the receiving software is determined by many factors, both in hardware and in software. To measure the one-way latency from the transmitter to the receiver is difficult without precise timing on both sides. If one assumes that the one-way latency is half of the round trip latency, then one can take advantage of the fact that measuring the round-trip latency is relatively easy in a bidirectional communications setting such as Bluetooth.

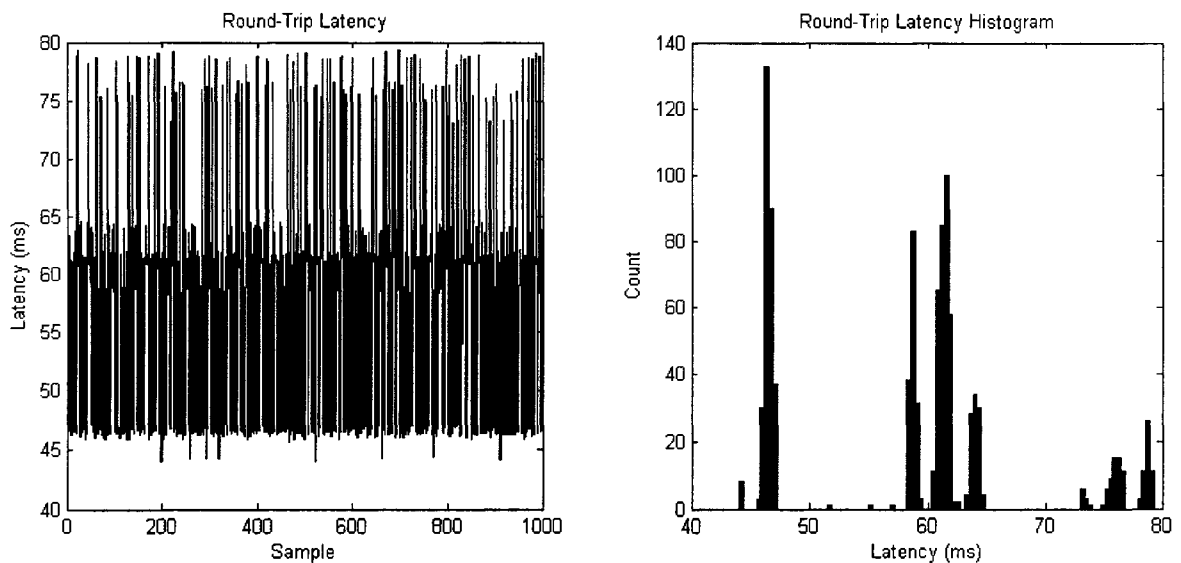


Figure 2.11: Round-trip latency of 1000 samples

To measure the round-trip latency with Smart Tool Holder, one simply sends a message from the PC and measures the time it takes to receive the reply. The PC can send a series of messages to get a better representation of the latency. The latency of 1000 messages and a histogram of these latencies is shown in Figure 2.11. This method of measuring latency does not account for the delay in the A/D converter. The delay

introduced by the converter, however, is less than 15 microseconds and can be reasonably ignored.

The average latency for these samples was 58.5ms, with a spread of 35.3ms. The standard deviation was 9.5ms. The histogram shows that the latency was not uniformly distributed, but rather tended to cluster around certain values. The exact cause of this phenomenon is unknown, but may be due to buffering of the data in the operating system, or the spacing of the Bluetooth transmission windows.

The latency of the transmitter meets the design requirement of 100ms even without the assumption of round-trip latency being twice the one-way latency. It should be noted, however, that this test only shows that the latency meets the design requirements for the test conditions, not that it necessarily does in all situations. Other PCs may have different latencies due to software or driver variations. To ensure that the latency meets the design requirements, one would have to use the device in conjunction with a PC utilizing a real-time operating system that can provide guarantees about timing.

2.4.4 Battery Discharge Characteristics

The discharge test monitored the voltage of the battery on the Smart Tool Holder under typical loading and transmitted the voltages to a PC. The discharge test puts the Smart Tool Holder software into battery monitoring mode, where instead of sending the torque value from the external A/D converter, it instead sends the voltage from the internal A/D converter connected to a resistor voltage divider across the battery (the electrical schematic in Appendix A gives full details on this voltage divider). The battery voltage can then be received by the PC instead of the torque data with no software

changes. In this way, the loading will be similar to the loading experienced during normal usage. The test used a freshly charged battery.

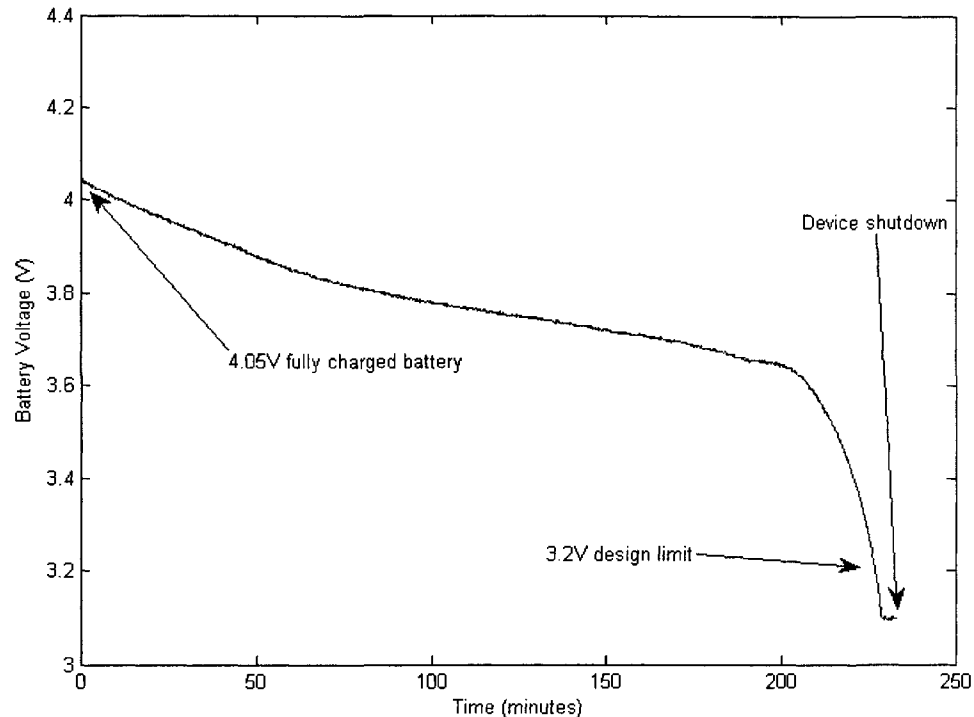


Figure 2.12: Battery discharge curve with typical load

The results of the discharge test are shown in Figure 2.12. It took 227 minutes to reach the design limit of 3.2V. The device continued to function for another 5 minutes before the lack of voltage caused the microcontroller to cease operations.

This test indicates that the selected battery can power the device for a much longer period of time than the stipulated hour. Additionally, it is possible to include another battery in the design, effectively doubling the run time.

2.4.5 Battery Charge Time

The charge test was to take batteries discharged to various levels, and measure the length of time it took the charging circuitry to fully charge them. Three tests were

performed, each at a different initial voltage. The results are summarized in Table 2.3.

Test #	Initial Battery Voltage (V)	Charging Time (min)
1	3.215V	88
2	2.833V	80
3	1.937V	93

Table 2.3: Battery charge times

While there is significant variation in the initial voltages, the charging times are relatively similar. The conclusion for the charging time is that it exceeds the design requirements. Even a completely empty battery can be fully charged in less than two hours.

2.5 Summary

Every design requirement, with the exception of the aliasing requirement, was met. Even though aliasing was present, it should only manifest itself as additional noise in the spectrum. The noise was experimentally verified as being $0.1010 \text{ N}\cdot\text{m}$. Though the noise was higher than expected, it still met the design requirement of being less than $0.15 \text{ N}\cdot\text{m}$.

The next chapter, Chapter 3, discusses the use of a new file format for storing experimental setup alongside cutting data for the purposes of archival. Readers interested solely in the Smart Tool Holder may skip ahead to Chapter 4 for a discussion of the applications of the Smart Tool Holder.

CHAPTER 3

INFORMATION TECHNOLOGY

3.1 Introduction

In machining research, data is acquired from a variety of sensors, analyzed, and then stored for later reference. In many situations, however, the data is never again used because either the particular experiment's exact setup has been forgotten or because the definition of the data format has changed (e.g. column 5 currently represents power but previously held audio information instead). This leads to cutting tests duplicating previous results, a waste of time and resources. Additionally, since the second experiment has no more documentation than the first, it is likely that future testing will again duplicate the original experiment.

The Extensible Markup Language (XML) [W3C XML] has successfully been used to facilitate data storage and exchange in machining systems through efforts such as MTConnect [MTConnect] and NCML [Ryou 2001, Ryou and Jerard 2001, Schuyler 2005, Jerard and Ryou 2006]. MTConnect focuses on collecting machine status from devices created by different vendors in a distributed environment. NCML was developed to provide a standard format for describing a part to be machined. These efforts have not, however, been targeted at the problem of annotating sensor data. While MTConnect can be used to collect data from low-bandwidth sensors, it is limited in what annotations can

be added. Using XML to annotate sensor data, has, however, been researched in other fields. WellLogML was developed to hold well logs for the oil industry [WellLogML]. Similarly in the medical field, ecgML was developed to hold electrocardiogram data [Wang, et al. 2003]. While neither of these solutions can be applied directly to machining, the principles and ideas that they are based on can be reused to create a machining sensor data storage format.

To be useful, a data format needs to be able to describe its contents explicitly and have the ability to change the format without invalidating older documents. Without these features, older data may become unreadable due to format changes. Additionally, the files should contain as much information as possible about the cutting experiment, so that the contents can remain useful after the specifics have been forgotten by the experimenters.

Because different research groups focus on describing different portions of the experimental setup, the format cannot be too specific about what must be present. Instead, it should provide a flexible method for research groups to add the information that is important for their research, without worrying about specifying the irrelevant data a broad format would impose.

XML was chosen for for the basis of the new file format because it can describe its contents explicitly and allows changes to the format that won't render older files useless. Because of its ubiquitous nature, many software tools and libraries already exist for reading and writing XML documents, saving considerable programming time during the development of analysis tools.

3.2 XML Schema

XML formats are each defined by an XML schema [W3C XML Schema]. A schema specifies the layout of the data within the files, and what data is optional or required. A graphical representation of the schema for the experiment results format is shown in Figure 3.1, and the full schema definition is contained within Appendix D. On the figure, the lines with diamonds represent a “is contained within” relationship. For example, an *experimenter* node is contained within a *personnel* node. An *experimenter* node cannot be located outside a *personnel* node. The numbers on the lines represent the cardinality of the relationship. A “1” means that the node is required to exist. A “0..1” means that the node is optional, but there cannot be more than one instance. A “0..*” means there can be any number of nodes, including none. A “1..*” means that at least one node is required to exist, but there may be any number of additional nodes.

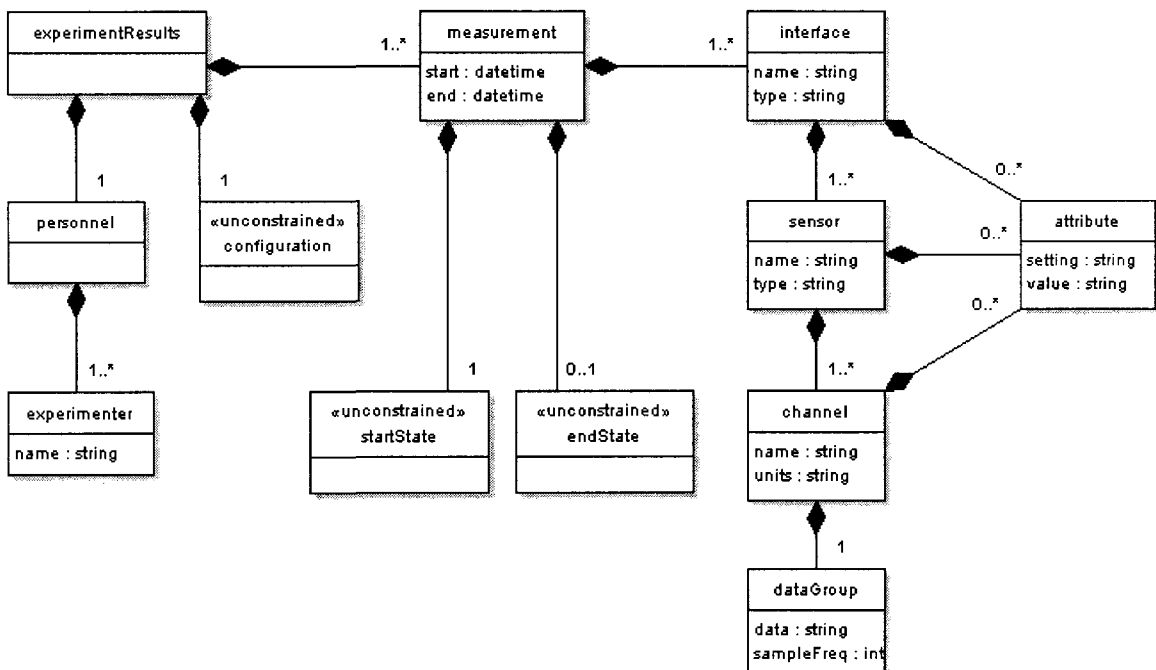


Figure 3.1: XML Schema for experiment results file

The contents of the blocks in Figure 3.1 represent required attributes of the nodes. For example, an *experimenter* node is required to have a name, and a *channel* node is required to have both a name and a unit. The <<unconstrained>> marked on *configuration*, *startState* and *endState* symbolize that the contents of those nodes are not defined by the schema, but rather are left up to each research group to determine for their specific needs.

Each of the nodes is described in detail:

- *experimentResults*: this is a container for everything else in the file, since XML only allows a single top-level node.
- *personnel*: this node is just a container for the *experimenter* nodes.
- *experimenter*: each of these nodes contain the name of a person associated with running the experiment. At least one *experimenter* node is required per file.
- *configuration*: this unconstrained node holds setup information relevant to the entire experiment. For example, the identity of the CNC machine would go in here, since that doesn't change while performing an experiment.
- *measurement*: conceptually, a measurement is a continuous stream of data from various sensor inputs over a short duration. If continuous data over a long duration is required, the data can be split across multiple measurements.
- *startState*: this unconstrained node holds setup information known at the beginning of the measurement, such as the bed position, tool type, or the G-code line being executed.
- *endState*: this node is the counterpart to *startState*, and holds any information that

has changed since the beginning of the measurement, such as bed position. Each measurement should be short enough so that any change between *startState* and *endState* can be easily understood or interpolated.

- *interface*: this is a container for *sensors*. Conceptually, an *interface* is an A/D converter, or some other connection between sensors and the recording computer.
- *sensor*: this represents a physical sensor, such as a microphone or dynamometer. Each sensor contains at least one *channel*. For example, a triaxial accelerometer would contain a three *channel* nodes representing the different axes of measurement.
- *channel*: this is a single independent stream of data from a sensor. Each *channel* holds a single *dataGroup* for its captured data.
- *dataGroup*: this node holds whitespace separated data points, sampled at the specified sampling frequency.
- *attribute*: these nodes can be used to annotate the *interface*, *sensor*, and *channel* nodes with specific information, such as the port on the A/D converted used to sample the data.

An example file conforming to this schema is shown in Appendix E.

The UNH Smart Machining System Platform [Xu 2007] was extended to record sensor data in the XML format and included annotations such as axial depth, spindle speed, radial engagement, and part program line number. The open source Xerces-C++ XML Parser [Xerces-C++] was utilized by the program to create the XML output.

3.3 MATLAB Toolbox

To facilitate analysis of collected machining data, a MATLAB toolbox was created for reading the contents of the XML data files with the schemas described in Section 3.2. The bulk of the toolbox was written in the Java programming language because of the extensive XML parsing functionality that it provides. The Java portion has all of the parsing functions and unit transformations. The unit transformations are done using the JSR-275 Measures and Units library [JSR-275]. The Java portion of the toolbox also provides a Graphical User Interface (GUI) that allows a user to load files and select specific *channel* nodes to examine. Each of the analysis tools is written in MATLAB “m code”. They call specialized routines to handle the conversion between the Java code and

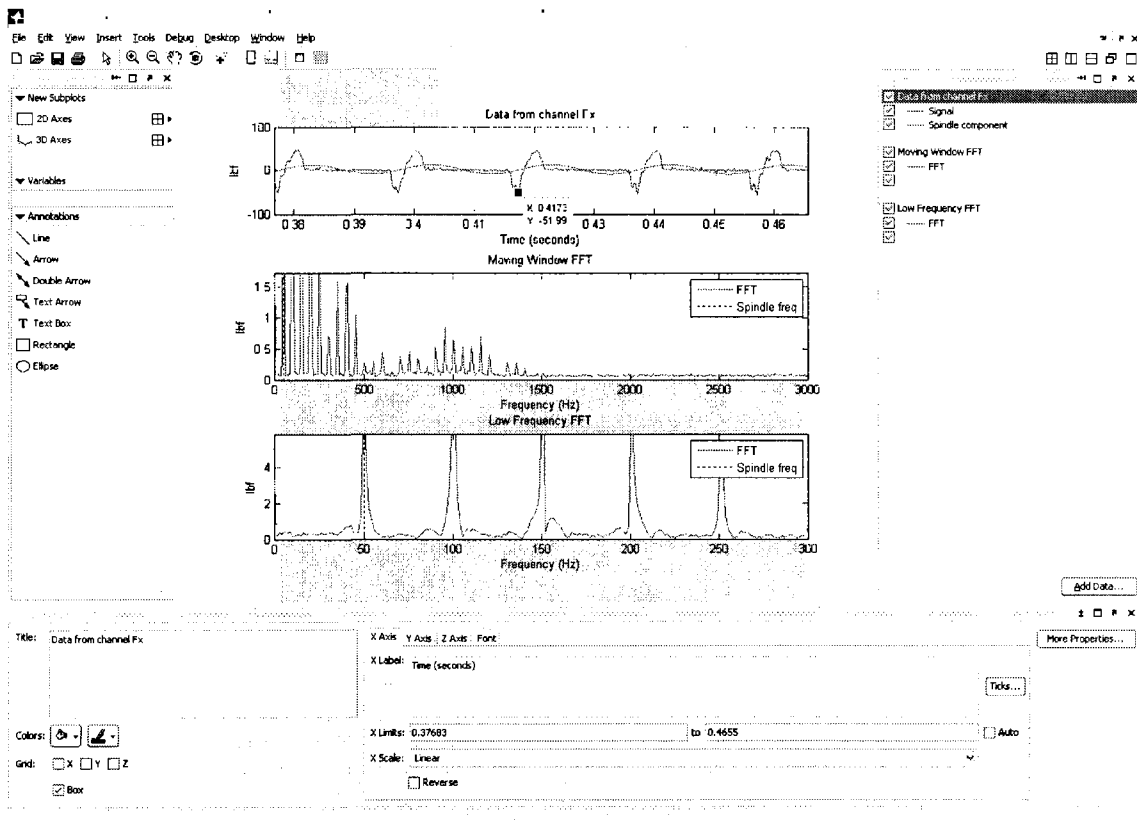


Figure 3.2: Example usage of the Fourier Analysis Tool

m code. Each of the analysis tools is written in m code, and calls the Java functions to load the data. Specialized routines transfer the data between the Java section of the toolbox and the m code section of the toolbox.

One of the tools contained in the MATLAB Toolbox is the *Fourier Analysis Tool*, which allows one to inspect both time and frequency domain representations of the channel data (see Figure 3.2). It uses the spindle speed annotation from the XML file to show the spindle frequency on the frequency domain plots, and provides an option to the user to add the spindle component to the time domain plot. All plots generated by the

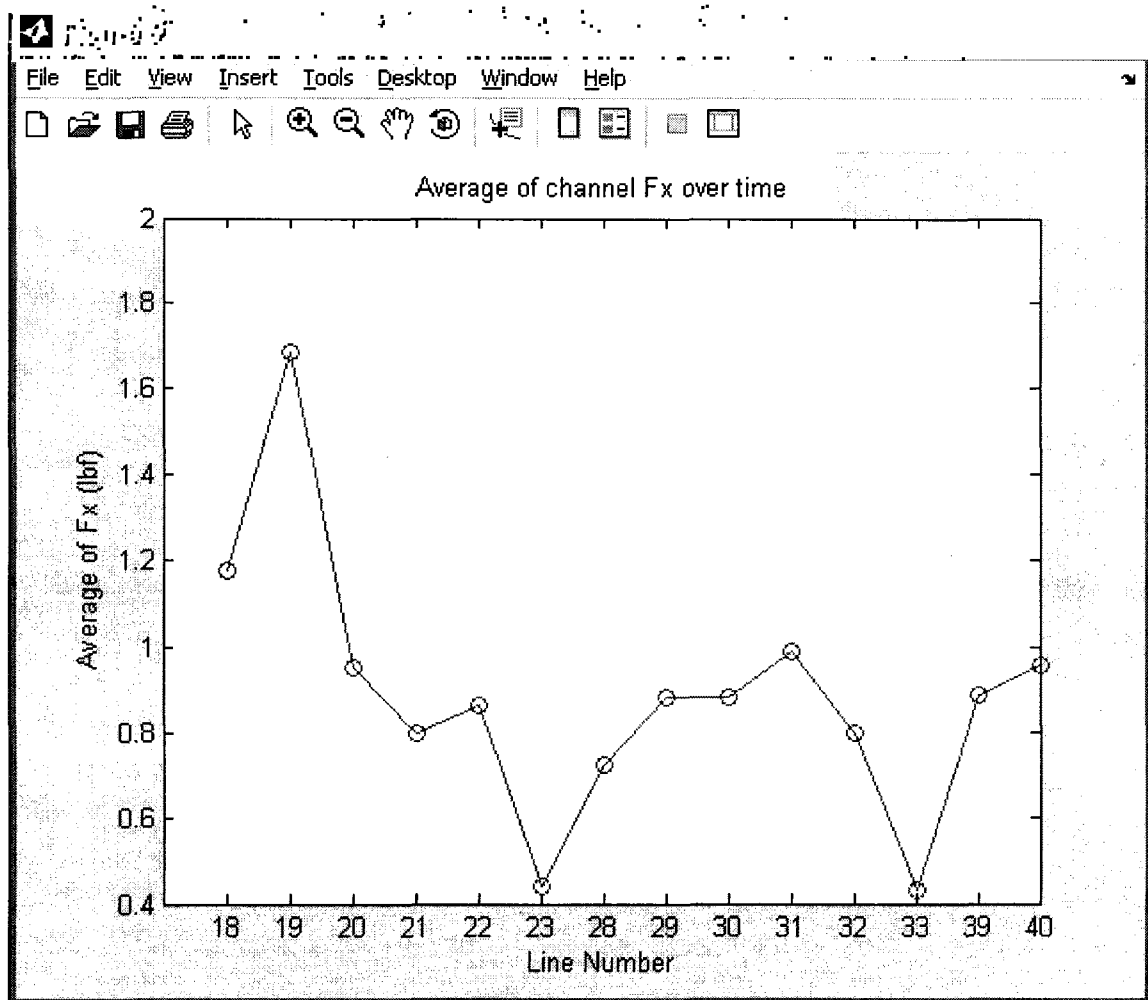


Figure 3.3: Example usage of the Average Plotting Tool

MATLAB Toolbox can be edited using the standard MATLAB tools and commands. For example, the figure axes can be relabeled to be more specific to the experiment. This provides maximum usability to the user while still allowing simple usage.

Another tool, the *Average Plotting Tool*, allows one to take multiple channels of data and plot their average. This is useful for visualizing the average power or force for different cutting conditions. An example plot is shown in Figure 3.3, where average force in the X axis is plotted versus the line number of the part program. For plotting mechanical power from the electrical power sensor data, the tool provides the user an option to subtract the tare power (as recorded in the XML file) and multiply by the motor efficiency. Power sensors are discussed further in Section 4.4. The *Average Plotting Tool* also provides the user with the option of plotting either the arithmetic mean or the Root Mean Squared (RMS) average. The arithmetic mean is given by:

$$x_{\text{arithmetic mean}} = \frac{1}{N} \sum_{i=1}^N x_i, \quad (3.1)$$

where x_i is the i th data point, and N is the total number of data points. The RMS average is given by:

$$x_{\text{RMS}} = \sqrt{\frac{1}{N} \sum_{i=0}^N x_i^2}. \quad (3.2)$$

The RMS average is useful with signals that have zero mean, such as the sinusoids found in AC voltage and current.

The *Ktc, Kte Regression Tool* is used to find the tangential coefficients K_{tc} and K_{te} of the cutting force model described in [Altintas 2000]. The tool takes power sensor data

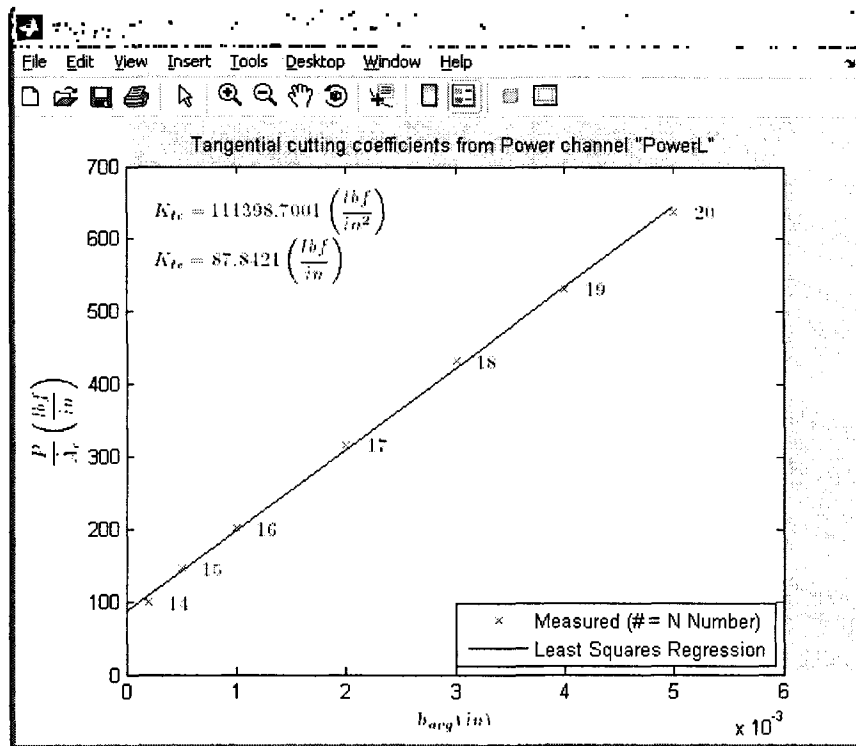


Figure 3.4: Example usage of the K_{tc} , K_{te} Regression Tool

from several cuts as its input, and plots P/\dot{A}_c (average mechanical power per swept contact area per time) versus h_{avg} (average chip thickness). This provides a way to estimate the force due to the chip thickness in a manner independent of cutting geometry. Each cut becomes a single data point on the plot. The slope of the trend line of these points is K_{tc} and the intercept is K_{te} (see Figure 3.4).

Further capabilities of the MATLAB Toolbox are described in Appendix F.

3.4 Summary

The XML based data files were used to hold the data from the wear tests performed in [Cui 2008] and [Javorek 2008], indicating that the file format is, at minimum, usable for controlled testing environments. The MATLAB Toolbox helped to simplify many repetitive analysis tasks.

CHAPTER 4

APPLICATIONS OF THE SMART TOOL HOLDER

4.1 Introduction

A high-bandwidth torque sensor like the Smart Tool Holder has direct application to the study of regenerative chatter. Regenerative chatter is a phenomenon in milling where surface “waviness” from small vibrations in the cutting tool left from the last tooth pass interfere with the vibrations from the current tooth pass, yielding even more waviness and vibration. This positive feedback can quickly lead to workpiece damage as the tool vibrations exceed the tolerances. Additionally, the high forces during chatter can destroy cutting tools and cause increased wear on spindle bearings. Chatter only occurs when the vibrations are out of phase with the vibrations from the last tooth pass, causing variation in the chip thickness. This means that chatter is dependent on the the tooth passing frequency (based upon the spindle speed and number of cutting teeth) and the chatter frequency. The chatter frequency is based upon the system properties (i.e. mass, stiffness and damping). The chatter frequency can be close to the natural frequency of the spindle and the tool, but may be influenced by the natural frequency of the workpiece, especially thin-walled parts.

Avoiding chatter can be accomplished by selecting spindle speeds that are integer divisors of the chatter frequency. At these stable speeds, the vibrations line up with the

vibrations from the last pass, and there is no variation in the chip thickness. The block diagram for a chattering milling system is shown in Figure 4.1. The tool and spindle compliance are represented by $G(s)$ and the cutting forces are represented by $F(s)$. The static chip thickness, h_0 , is the input to the system. The regeneration is provided by the time delay between teeth, with the current tooth removing the waviness left from the previous tooth. The cutting forces, represented by $F(s)$, are directly proportional to the axial depth of cut, a . This is the controlling term for the buildup of chatter. If the axial depth is greater than the critical value a_{lim} then the system is unstable and waviness will increase from the previous pass and chatter will develop. Conversely, if the axial depth is less than a_{lim} then chatter will not develop and will recede if already present.

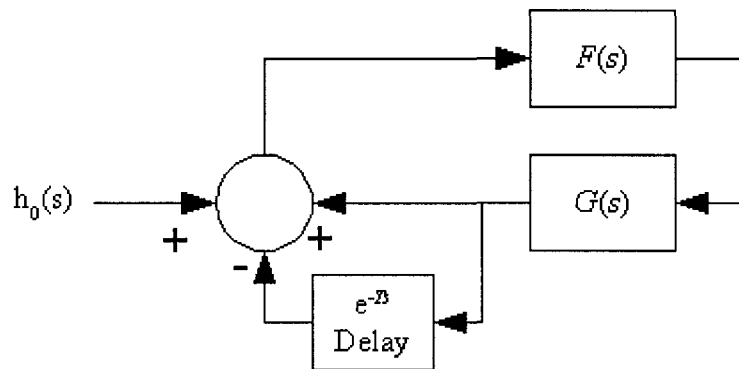


Figure 4.1: Block diagram of chattering milling system

One of the techniques used to predict chatter is to estimate the frequency response $G(s)$ of the spindle and tool from a hammer test [Altintas 2000, pp. 89], and use the frequency response to generate a stability lobe diagram. Stability lobe diagrams are a plot of a_{lim} versus spindle speed (see Figure 4.2). From this diagram, it is possible to select a stable speed for the desired axial depth, a , during toolpath generation. The problem using hammer tests is that they are prone to errors such as double taps. Additionally, the

frequency response of the stationary spindle may not be the same as the frequency response of the rotating spindle [Abele and Fiedler 2004].

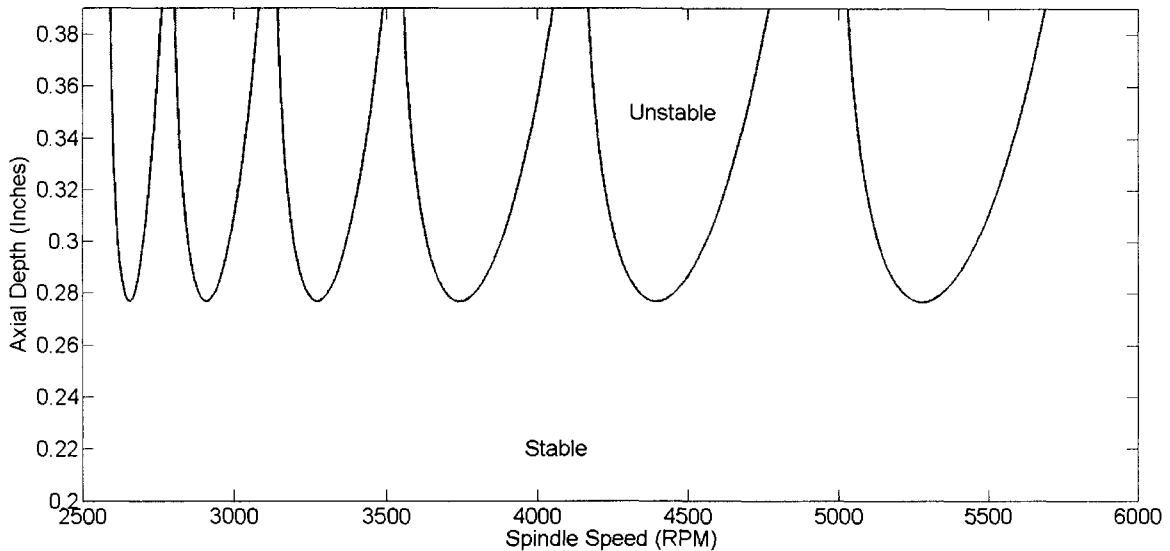


Figure 4.2: Example stability lobe diagram [Suprock, et al. 2008]

The Harmonizer [Delio 1992] uses a directional air microphone to capture auditory data during chatter conditions and compute stable spindle speeds based upon the chatter frequency present in the data. In this manner, the problem of the spindle having a different response while stationary is avoided, since all of the measurement happens during cutting conditions. This method only works if chatter is present and suffers from the fact that the microphone can pick up unrelated background noise.

Both of these solutions assume that the frequency response of the system is unchanging. However, in many cases, such as during machining of flexible or thin-walled parts, the frequency response changes during the cutting process. This can make selection of spindle speeds during toolpath generation extremely difficult, since the dynamics of an ever changing workpiece must be simulated to determine stable spindle speeds. An alternative is to use an on-line chatter controller to analyze the system and adjust the

spindle speed during the cutting process [Ismail and Ziaei 2002].

The Smart Tool Holder has the potential to be useful in both off-line and on-line situations. Because the Smart Tool Holder measures torque rather than sound pressure through a microphone, there is significantly less interference from background noise. Furthermore, it may be possible to determine chatter frequencies without the presence of actual chatter [Suprock, Jerard, Fussell 2009].

This chapter discusses both the foundations and implementation of a time-domain model to simulate on-line chatter controllers. A new model is required in order to simulate the transients during changing spindle speeds. This chapter also discusses the use of the Smart Tool Holder as a power sensor to overcome some of the limitations of conventional power sensing technology.

4.2 Force and Torque Model for a Chatter Simulator

To simulate chatter for the evaluation of various chatter control techniques, it is necessary to create a model for the tool forces based on the displacement of the tool from the previous tooth pass. Because chatter controller simulations involve changing spindle speeds, it is necessary to develop a formulation of a cutting force model that does not depend upon the tool rotating at a fixed spindle speed. Transients related to the changing RPM can therefore be simulated, and spindle speed controllers can be more accurately evaluated. It is also desirable to formulate the model so that as many calculations can be done at the start of the simulation to lessen the computational burden and reduce the overall length of the simulation run.

In addition to the fixed spindle speed assumption, some of the existing chatter

models assume a cutter with zero helix angle to simplify the analytical force equations [Altintas 2000]. In this model, adding the helix angle term to the simulation does not significantly alter the computational burden of the simulator, since the force coefficients that depend on the helix angle are precomputed. Additionally, this model includes edge coefficients K_{le} , K_{re} , and K_{ae} to study how they affect the build up of chatter. These edge coefficients are sometimes ignored in chatter simulations producing stability lobe diagrams, since the simulations only deal with constant cutting geometry [Schmitz and Smith 2009, pp. 161]. Including them, however, can give a better estimate of what happens across a wide range of cutting conditions, and including them does not significantly increase the necessary computations. This model also includes terms to simulate tool runout, a condition where the rotational center of the tool is not the same as the geometric center. Runout can hinder the accurate detection of chatter in cases where the chatter frequency is a multiple of the spindle frequency but not a multiple of the tooth passing frequency [Schmitz and Smith 2009, pp. 213].

This model makes the assumption that all teeth are evenly spaced along the circumference of the tool. It would be possible to extend the simulator to support uneven tooth spacing, but it would require storing additional data about the surface after each tooth pass. This model also assumes that the angle of deflection is very small (or stated another way, that displacements are uniform with height). The angle of deflection is given by the following equation:

$$\theta = \frac{F l^2}{2 E I}, \quad (4.1)$$

where F is the force acting in the radial direction, l is the length of the tool, E is the

elastic modulus of the tool material, and I is the area moment of inertia. This means that the assumption of very small angular deflection is valid for stiff tools and low forces. This model also assumes that the path the cutting teeth follow is a circular one, rather than the true cycloidal path. The circular path is a close approximation except for the part of the cut with shallow radial engagements (these engagements correspond to very low forces, and therefore should introduce only minimal error into the chatter simulation). Another assumption is that the displacement due to chatter never exceeds the chip thickness. This means that tooth cannot “jump out of the cut”.

4.2.1 Force Model

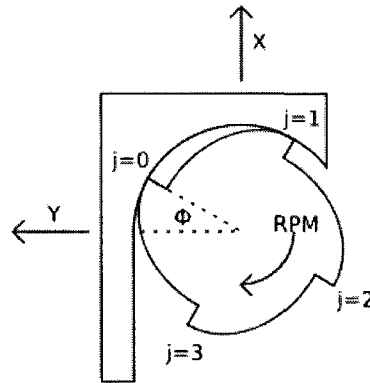


Figure 4.3: The angle of engagement

For a helical flat endmill with N evenly distributed teeth, the angle of engagement (see Figure 4.3) for a point at height z along the cutting edge on tooth j is defined as [Altintas 2000, pp. 43]:

$$\phi_j(\phi, z) = \phi + \frac{j * 2 \pi}{N} - \frac{z * \tan(\alpha_{helix})}{R(z)}; j = 0, 1, \dots, N - 1, \quad (4.2)$$

where ϕ is the angle of engagement for the bottom most point of tooth $j=0$, R is the radius of the cutter as a function of z , and α_{helix} is the helix angle (positive helix angles

mean that the bottom most point of the tooth is the first portion to enter the material).

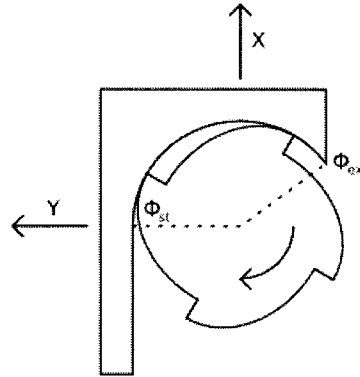


Figure 4.4: Angular limits of cutting

For simple cutting geometries with constant entrance and exit angles, the following function g_j determines if a point on the cutting edge is in the material, where ϕ_{st} is the start angle for cutting and ϕ_{ex} is the exit angle for cutting (see Figure 4.4). The tool is moving in the positive direction along the X-axis. The function g_j takes on the value 1 if the point on the tooth is in the cut, and 0 if it is not in the cut. This allows the function to be used as a multiplier to limit the forces to the interior of the cut:

$$g_j(\phi, z) = \begin{cases} 1 & \text{point is in the cut} \\ 0 & \text{otherwise} \end{cases} = \begin{cases} 1 & \phi_{st} \leq \phi_j(\phi, z) \leq \phi_{ex} \\ 0 & \text{otherwise} \end{cases}. \quad (4.3)$$

The engagement angle ϕ_j must be wrapped into the range $[0, 2\pi]$ for the comparisons.

The chip thickness, h , is defined as the thickness of the workpiece material being cut that exists in the radial direction. On a tool with evenly spaced teeth, the chip thickness is equal to the radial component of the displacement from the cutting edge of the previous tooth when it was at the same angular engagement. The displacement comes in two parts: the dynamic displacement (Δx and Δy , see Figure 4.5) and the static displacement, or feed per tooth (f_{pt}). The feed per tooth is a function of the bed feed rate,

spindle speed and the number of teeth:

$$f_{pt} = \frac{feed}{N * f_{spindle}} = \frac{feed * 60}{N * RPM} \quad (4.4)$$

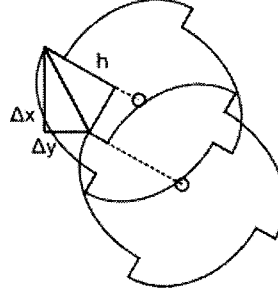


Figure 4.5: Chip thickness due to dynamic displacement

The fact that the chip thickness is the projection of the displacement in the radial direction rather than the entire magnitude of the displacement allows the function to be written as a coordinate transformation from Cartesian to the radial axis. The chip thickness can be written using the functions defined in equations (4.2) and (4.3):

$$h_j(\phi, z) = g_j(\phi, z) [(\Delta x + f_{pt}) \sin(\phi_j(\phi, z)) + \Delta y \cos(\phi_j(\phi, z))]. \quad (4.5)$$

Making the following definitions:

$$h_{x,j}(\phi, z) = g_j(\phi, z) \sin(\phi_j(\phi, z)), \quad (4.6)$$

$$h_{y,j}(\phi, z) = g_j(\phi, z) \cos(\phi_j(\phi, z)), \quad (4.7)$$

allows the chip thickness equation to be written in a more compact form:

$$h_j(\phi, z) = \begin{bmatrix} h_{x,j}(\phi, z) & h_{y,j}(\phi, z) \end{bmatrix} \begin{bmatrix} \Delta x + f_{pt} \\ \Delta y \end{bmatrix}. \quad (4.8)$$

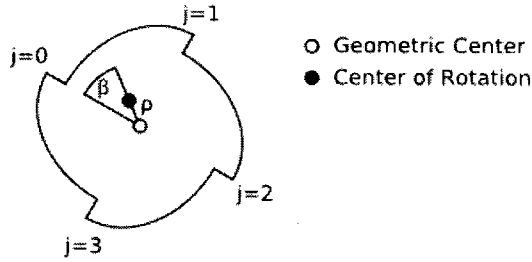


Figure 4.6: Tool runout magnitude and locating angle

The equations (4.5) and (4.8) are sufficient for tools with no runout, but must be altered slightly to account for cases with rotational eccentricities. The magnitude of the runout is ρ and the locating angle from tooth $j=0$ is β (see Figure 4.6). This leads to the equation for runout at any point along cutting edge j at height z :

$$\rho_j(z) = \rho * \cos[\beta - (\phi_j(\phi, z) - \phi)] \quad (4.9)$$

Note that this is independent of the rotation angle ϕ of the tool. This equation makes the assumption that the magnitude of the runout is much, much smaller than the radius of the cutter. This allows any change in the tooth locating angles to be ignored.

Using the following definition for the invariant portion of chip thickness:

$$h_{c,j}(\phi, z) = g_j(\phi, z) (\rho_j(z) - \rho_{j-1}(z)) \quad (4.10)$$

The equation (4.8) can be extended to include the effects of runout:

$$h_j(\phi, z) = \begin{bmatrix} h_{x,j}(\phi, z) & h_{y,j}(\phi, z) & h_{c,j}(\phi, z) \end{bmatrix} \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} \quad (4.11)$$

It should be noted that this is only an approximation of the chip thickness, as these equations may yield negative values of chip thickness for large dynamic displacements, something that is not possible in the physical system.

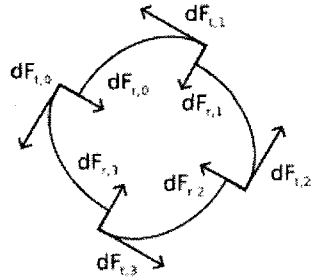


Figure 4.7:
Differential forces on
a slice of a cutting tool

The tangential and radial forces (see Figure 4.7) on any differential piece are determined by the chip thickness, h , and the six coefficients, K_{tc} , K_{te} , K_{rc} , K_{re} , K_{ac} , and K_{ae} . These coefficients depend on the material, tool geometry, and sharpness of the tool [Altintas 2000, pp. 36]. The cutting coefficients K_{tc} , K_{rc} , and K_{ac} give forces based upon the chip thickness, whereas the edge coefficients K_{te} , K_{re} , and K_{ae} represent rubbing and plowing, and are simply added whenever the differential section of the cutter is inside the material. The forces on this differential piece are:

$$\begin{bmatrix} dF_{t,j}(\phi, z) \\ dF_{r,j}(\phi, z) \\ dF_{a,j}(\phi, z) \end{bmatrix} = \begin{bmatrix} K_{tc}h_j(\phi, z) + K_{te}g_j(\phi, z) \\ K_{rc}h_j(\phi, z) + K_{re}g_j(\phi, z) \\ K_{ac}h_j(\phi, z) + K_{ae}g_j(\phi, z) \end{bmatrix} dz. \quad (4.12)$$

This equation can be rewritten to separate the cutting coefficients and chip thickness components into their own matrices:

$$\begin{bmatrix} dF_{t,j}(\phi, z) \\ dF_{r,j}(\phi, z) \\ dF_{a,j}(\phi, z) \end{bmatrix} = \begin{bmatrix} K_{tc} & K_{te} \\ K_{rc} & K_{re} \\ K_{ac} & K_{ae} \end{bmatrix} \begin{bmatrix} h_{x,j}(\phi, z) & h_{y,j}(\phi, z) & h_{c,j}(\phi, z) \\ 0 & 0 & g_j(\phi, z) \end{bmatrix} \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} dz, \quad (4.13)$$

$$\begin{bmatrix} dF_{t,j}(\phi, z) \\ dF_{r,j}(\phi, z) \\ dF_{a,j}(\phi, z) \end{bmatrix} = \mathbf{K} * \mathbf{H}_j(\phi, z) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} dz. \quad (4.14)$$

The tangential and radial forces along the differential piece can be transformed into Cartesian coordinates using matrix multiplication:

$$\begin{bmatrix} dF_{x,j}(\phi, z) \\ dF_{y,j}(\phi, z) \\ dF_{z,j}(\phi, z) \end{bmatrix} = \begin{bmatrix} -\cos(\phi_j(\phi, z)) & -\sin(\phi_j(\phi, z)) & 0 \\ \sin(\phi_j(\phi, z)) & -\cos(\phi_j(\phi, z)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} dF_{t,j}(\phi, z) \\ dF_{r,j}(\phi, z) \\ dF_{a,j}(\phi, z) \end{bmatrix}, \quad (4.15)$$

$$\begin{bmatrix} dF_{x,j}(\phi, z) \\ dF_{y,j}(\phi, z) \\ dF_{z,j}(\phi, z) \end{bmatrix} = \mathbf{A}_j(\phi, z) \begin{bmatrix} dF_{t,j}(\phi, z) \\ dF_{r,j}(\phi, z) \\ dF_{a,j}(\phi, z) \end{bmatrix}. \quad (4.16)$$

These differential Cartesian forces can then be integrated in z from 0 to a , the axial depth of cutting, to give the total forces along the cutting tooth j :

$$\begin{bmatrix} F_{x,j}(\phi) \\ F_{y,j}(\phi) \\ F_{z,j}(\phi) \end{bmatrix} = \int_0^a \begin{bmatrix} dF_{x,j}(\phi, z) \\ dF_{y,j}(\phi, z) \\ dF_{z,j}(\phi, z) \end{bmatrix} = \int_0^a \left(\mathbf{A}_j(\phi, z) * \mathbf{K} * \mathbf{H}_j(\phi, z) * \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} dz \right). \quad (4.17)$$

Because the displacements do not depend upon z (this was one of the model's original assumptions), they can be moved outside the integration:

$$\begin{bmatrix} F_{x,j}(\phi) \\ F_{y,j}(\phi) \\ F_{z,j}(\phi) \end{bmatrix} = \left(\int_0^a \mathbf{A}_j(\phi, z) * \mathbf{K} * \mathbf{H}_j(\phi, z) dz \right) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} = \mathbf{F}_j(\phi) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix}. \quad (4.18)$$

The total forces upon the tool are given by the sum of the forces on each tooth:

$$\begin{bmatrix} F_x(\phi) \\ F_y(\phi) \\ F_z(\phi) \end{bmatrix} = \left(\sum_{j=0}^{N-1} \mathbf{F}_j(\phi) \right) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} = \mathbf{F}(\phi) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix}. \quad (4.19)$$

The \mathbf{F} matrix can be used to find the forces on the tool using the engagement angle ϕ , and the displacement from the last tooth pass. The \mathbf{F} matrix can be precomputed for a

complete revolution of angles since it is independent of the displacements being simulated. Typically, the integration to find the \mathbf{F} matrix is approximated by a summation over discrete slices of z . An example program for computing this matrix is shown in Appendix G.

4.2.2 Torque Model

The torque on a differential piece of the tool comes from the radius, R , and the tangential force upon that differential piece:

$$dT_j(\phi, z) = R(z) * dF_{t,j}(\phi, z) = R(z) * [K_{tc} \quad K_{te}] * \mathbf{H}_j(\phi, z) * \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} dz. \quad (4.20)$$

This differential torque can be integrated through the Z-axis to get the total torque on the cutting tooth:

$$T_j(\phi) = [K_{tc} \quad K_{te}] * \int_0^a (R(z) * \mathbf{H}_j(\phi, z) dz) * \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix}. \quad (4.21)$$

This can be consolidated into a single matrix, \mathbf{T}_j :

$$T_j(\phi) = \mathbf{T}_j(\phi) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix}. \quad (4.22)$$

Then the entire torque on the tool can be determined by summing the torques on the individual teeth:

$$T(\phi) = \left(\sum_{j=0}^{N-1} \mathbf{T}_j(\phi) \right) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix} = \mathbf{T}(\phi) \begin{bmatrix} \Delta x + fpt \\ \Delta y \\ 1 \end{bmatrix}. \quad (4.23)$$

As in the case of the forces, the \mathbf{T} matrix can be precomputed for various values

simulation takes RPM, feedrate, and force noise as inputs, and outputs forces in the Cartesian coordinates, as well as torque. The force noise is additive, and is intended to be used to study the effect of random noise on the feedback loop. There are three main pieces to the simulator: the *Constant Geometry Cutting Forces* block, the *Spindle Compliance* block, and the *Variable Transport Delay* block.

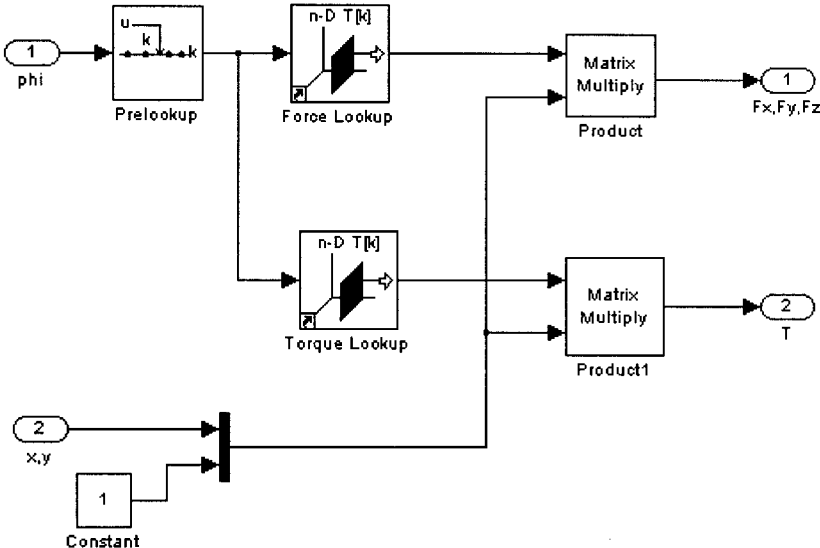


Figure 4.9: Simulink diagram for cutting forces from displacement

The *Constant Geometry Cutting Forces* block implements a lookup table for the \mathbf{F} and \mathbf{T} matrices from Section 4.2. It uses the program shown in Appendix G to compute the matrices before simulation commences. The details of the implementation for this block are shown in Figure 4.9. It uses the cutting angle ϕ to determine the cutting forces F_x , F_y , and F_z for the given displacement (both static and dynamic).

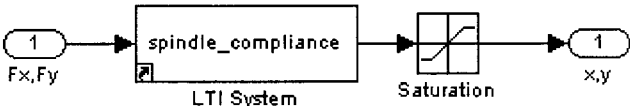


Figure 4.10: Simulink diagram for displacement from cutting forces

The *Spindle Compliance* block, takes the F_x and F_y output from the *Constant Geometry Cutting Forces* block and, using a two by two transfer function matrix representing the tool and spindle, creates a displacement in x and y . The details of the Simulink implementation are shown in Figure 4.10. The use of a saturation block ensures that displacements do not reach unrealistic level during the exponential growth of chatter.

The *Variable Transport Delay* block stores the displacement from the previous tooth pass. The main input to this block is the output of the *Spindle Compliance* block. The other input is the delay per tooth, computed from the RPM. This block does the work necessary to correctly store the displacement for the previous tooth pass, even during spindle speed changes. Using the output from this block the dynamic displacement can be easily calculated. This is done by taking the displacement for current tooth pass (The output of the *Spindle Compliance* block) and subtracting the displacement for the previous tooth pass (the output from the *Variable Transport Delay* block). The complete displacement can be found by adding the feed per tooth (in the x direction, only) to the dynamic displacement. The complete displacement forms one of the two inputs to the *Constant Geometry Cutting Forces* block (the other is the rotational angle ϕ).

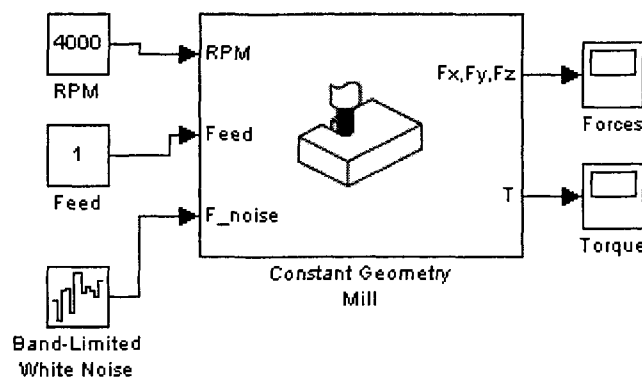


Figure 4.11: Basic usage for chatter simulator

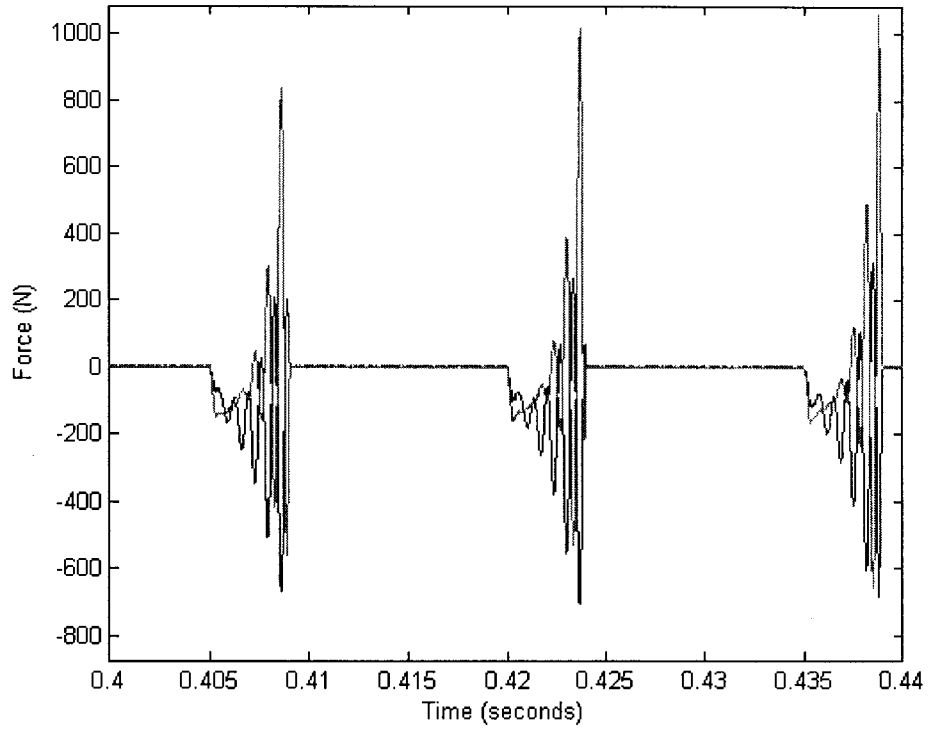


Figure 4.12: *X and Y forces for a simulated cut at 4000 RPM (chatter)*

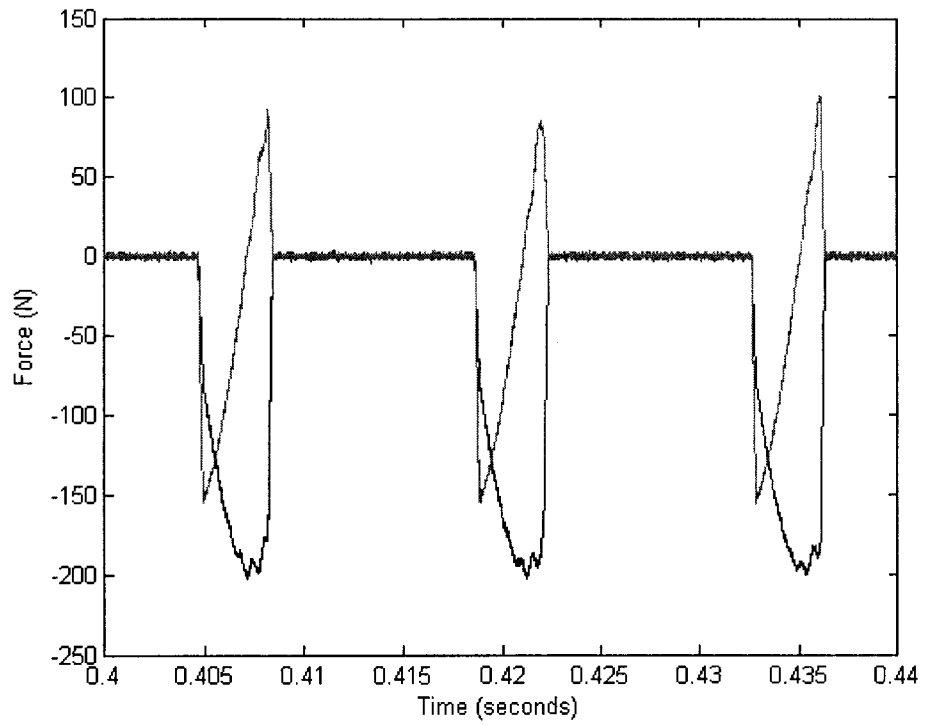


Figure 4.13: *X and Y forces for a simulated cut at 4300 RPM (stable)*

A basic usage of these components is shown in Figure 4.11. The output of the simulator is shown in Figure 4.12 and Figure 4.13. The only difference between these two runs was the spindle speed. The cut was unstable at 4000 RPM, and stable at 4300 RPM. The cutting conditions for these runs were: 1 tooth, 30° helix angle cutter with 19.05mm diameter, half immersion upmill, axial depth of 3.5mm, feed rate of 1mm/sec, $K_{te} = 650$, $K_{te} = 20$, $K_{rc} = 300$, $K_{rc} = 45$. Figure 4.14 shows a half immersion downmill with a 4-tooth cutter, axial depth of 1mm, and spindle speed of 4600. This figure is meant to illustrate the wide range of cuts that can be simulated.

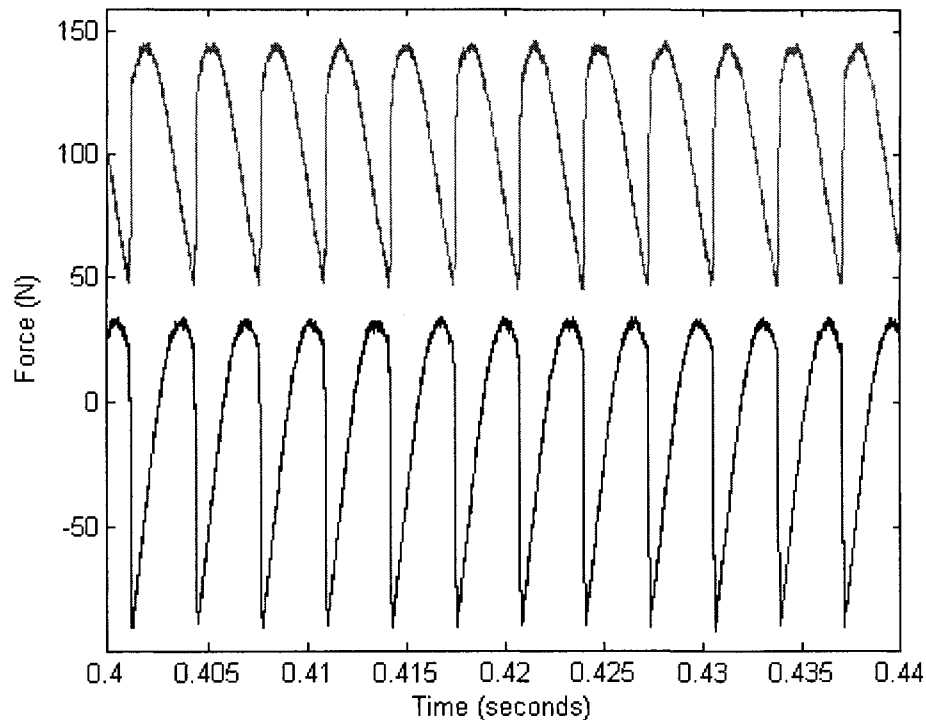


Figure 4.14: X and Y forces for a simulated downmill using a 4-tooth cutter

The simulator was designed to show the response of various chatter controllers, but presently there are no controllers that have been implemented in Simulink. This is part of the future work for the simulator. A complete discussion of future work is

presented in Chapter 5.

4.4 Power Sensor

Spindle power sensors are non-invasive sensors that can be used to acquire an estimate of the average mechanical cutting power. From the average power, the tangential model coefficients, K_{tc} and K_{te} , can be estimated using linear regression. These tangential model coefficients can be used to track wear [Desfosses 2007, Cui 2008] and can be used during process planning to optimize feedrate selections [Jerard, et al. 2006]. Spindle power sensors, however, have some limitations. To accurately estimate the mechanical power, the spindle motor's efficiency curve must be known. Determining this curve is an expensive and time-consuming process. Furthermore, the efficiency can change over the lifetime of the spindle as the bearings wear. Spindle power sensors must also subtract the unloaded power, or tare power, from the measured power. The tare power, however, is temperature dependent. This means it can change with the motor load, complicating the accurate measurement of power.

The Smart Tool Holder overcomes these limitations for the estimation of mechanical power by using the cutting torque. Equation (2.1), repeated here, gives the relationship between mechanical power and torque:

$$P(t) = T(t) \omega = T(t) \frac{2\pi \text{RPM}}{60} . \quad (4.24)$$

Since electrical power is not used, both the motor efficiency and tare power are unneeded. Instead, the RPM of the spindle must be known, but this is not typically an issue.

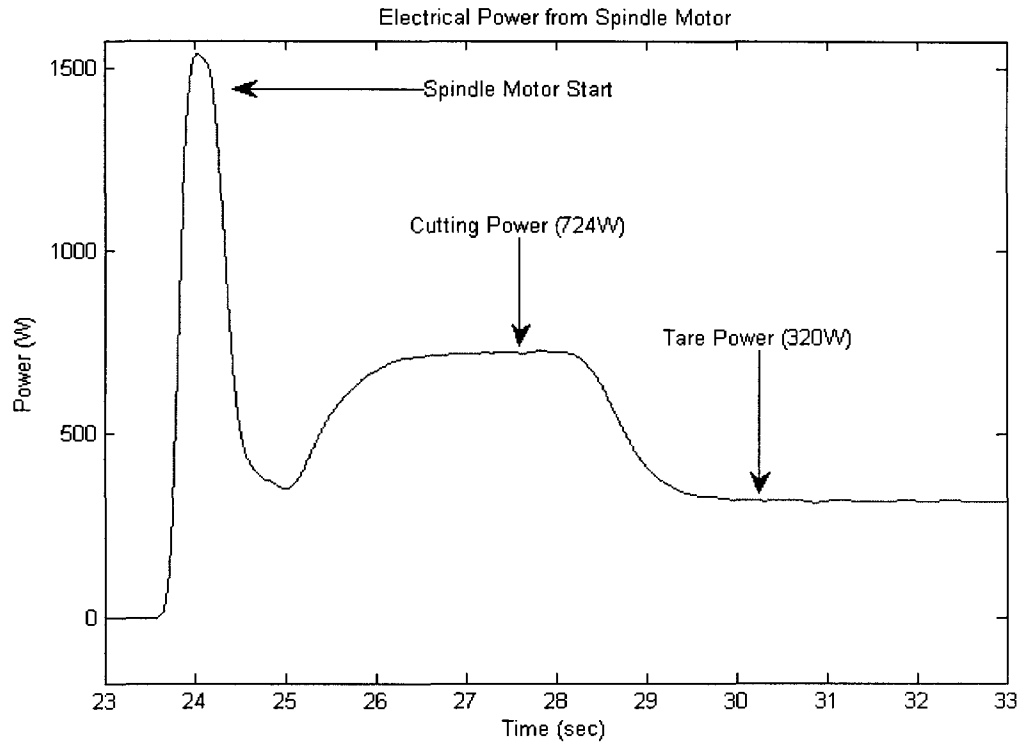


Figure 4.15: Power sensor data (filtered for clarity)

Figure 4.15 and Figure 4.16 show the results from a test cut for both the power sensor and the Smart Tool Holder (note that due to data acquisition limitations these figures are not aligned in time). The power sensor measured 724W during cutting with 320W of tare power, leaving 404W expended to perform cutting. All measurements are real power and do not contain reactive power. Using the spindle motor efficiency curve from [Xu 2007] gives an approximate efficiency of 93%. This gives 375.7W as the estimated mechanical power.

The signal from the Smart Tool Holder gives an average torque of 1.599 N·m . Using (4.24) with a spindle speed of 2501RPM gives an average power estimate of 418.8W. This is a difference of approximately 10%. The source of this difference is

unknown.

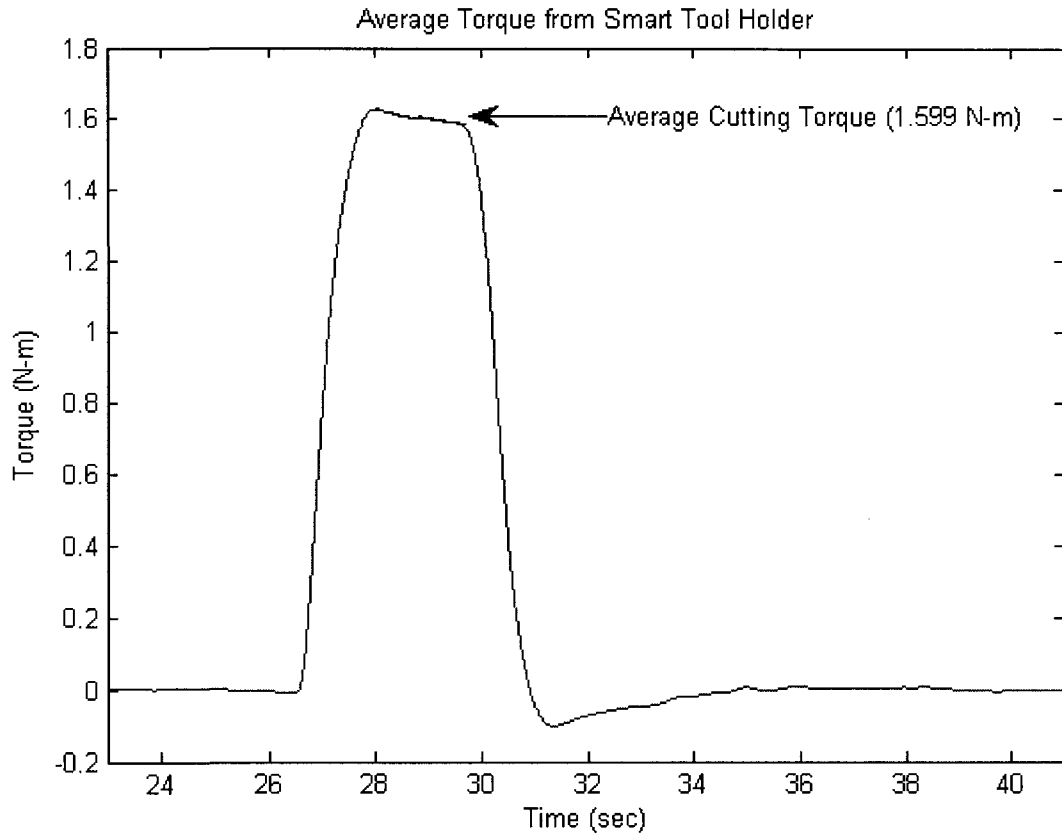


Figure 4.16: Smart Tool Holder data (filtered to show average torque)

The small dip in Figure 4.16 after cutting is due to temperature effects on the strain gauge bridge. The cutting process generates a temperature gradient across the strain gauge bridge, leading to a DC offset. This offset can reduce the accuracy of a power measurement (or any measurement dependent on the average torque). A “tare torque” solution is possible, but one of the goals of the Smart Tool Holder was to overcome the tare power limitation of conventional power sensors. Further solutions to the offset issue are discussed as future work in Section 5.2.5.

4.5 Summary

The simulator developed for testing control strategies has the ability to chatter and

to model transients related to changing spindle speeds. No chatter controller was developed or tested during this research. The development and testing are discussed in Section 5.2.8 as future work.

Though there was only a partial match between the between the mechanical power estimate of the Smart Tool Holder and the estimate from a conventional spindle power sensor, the Smart Tool Holder does have potential as a cutting power sensing device. Future analysis may provide insight about the source of the difference between the two estimates.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

The Smart Tool Holder meets nearly all of the design requirements laid down for a torque sensor. The one requirement not met, sampling at a rate high enough to avoid aliasing, only impacts normal usage with higher than expected noise levels. Methods to overcome this limitation are discussed in Section 5.2.

The XML file format proposed in Section 3.2 was utilized for the data collection system in the Design and Manufacturing Laboratory at UNH. Working with the XML output files was accomplished through the use of the developed MATLAB Toolbox. This shows feasibility.

The chatter simulator developed in Section 4.3 was capable of simulating chatter and transients relating to changing spindle speed. This simulator will be helpful in the development of a real-time feedback control system for chatter suppression utilizing the Smart Tool Holder. The development of such a controller is discussed in Section 5.2.8 as future work.

The ability of the Smart Tool Holder to estimate mechanical power was also experimentally evaluated. There was a 10% difference between the output of the Smart Tool Holder and the output of the power sensor. Estimating mechanical power directly

from torque has the potential to overcome many of the limitations of spindle motor power sensors.

5.2 Future Work

The aliasing limitation on the current Smart Tool Holder has several possible solutions. The main problem is that the sampling rate is limited by the transmission rate of the radio system, since all of the data is being sent to the PC. One solution to the aliasing problem is to change the anti-aliasing filter to have a steeper roll-off. Another is to increase the transmission rate of the radio module, and sample more quickly. Another solution is to process the data before it's sent to the computer, using the available transmission capacity more efficiently. This could be done by performing the analysis directly on the tool holder's processor and sending the results to the PC, or by utilizing a data compression scheme to send the full data set.

5.2.1 Increase the Anti-Aliasing Filter Order

The current anti-aliasing filter is not sufficient to prevent aliasing within the passband. Future work would be to determine the best filter order and transfer function to effectively cut out the unwanted frequencies, and implement it in hardware. Things to consider when implementing in hardware are filter topology, physical size, component cost, and sensitivity to component variation.

5.2.2 Replace Bluetooth Radio With Another Technology

The Roving Networks RN-41 Bluetooth Module limits the data rate to 260kbps throughput. The constraint on throughput limits the sampling rate to 16kHz, unless some form of data compression is utilized. Other radio systems are available with greater

throughput rates. The nRF24L01+ produced by Nordic Semiconductor [Nordic Semiconductor] is a low cost device that advertises 2Mbps transmission rate, though a portion of that rate is consumed by overhead.

Replacing the RN-41 with the nRF24L01+ has other advantages as well: it consumes less power, has a smaller footprint, and less connection delay. The RN-41 consumes a large portion of the power in the current system (see Section 2.3.9). The nRF24L01+ consumes 13.5mA peak, compared to the 100mA peak and 30mA average of the RN-41. The smaller footprint of the nRF24L01+ and its necessary external components as compared to the current RN-41 Bluetooth Module has the potential to reduce the size of the main circuit board. Finally, the RN-41 has a delay on the order of 10 seconds when first initializing the connection to the PC. The delay is not a problem for research usage, but could be an issue in industrial settings where tool changes can happen in less than 2 seconds. The simpler protocol used by the nRF24L01+ facilitates a quicker connection establishment. The drawback to using an alternate radio system is that it would not be compatible with the wide array of PC Bluetooth dongles that are commercially available.

Future work includes redesign of the main electronics to use a nRF24L01+ instead of the RN-41. This also involves rewriting portions of the Smart Tool Holder's program, since the nRF24L01+ uses a different protocol than the existing radio. The new design would also require PC compatible receiving hardware, and any associated software drivers.

5.2.3 Process Data On-Board

An alternative to increasing the transmitter bandwidth is to reduce the amount of data needing transmission. Processing the data on board can reduce the size of the data by either using data compression or by analyzing the data directly. Data compression could be either lossless or lossy. If a lossy compression is used, careful attention should be paid to examining how the distortion affects the analysis algorithms. Analyzing the data directly would allow the device to transmit only the final output of the algorithm, saving a considerable amount of bandwidth.

The current ATmega164P processor already spends a large portion of its time sampling the data, arranging it for transmission, and formatting it for display on the LED bar graph. To implement compression or analysis would almost certainly require a faster processor. There are many potential processors that balance the need for computation with low power consumption. Depending on the exact structure of the analysis algorithms, a processor with Digital Signal Processing (DSP) commands could be useful to reduce the number of program instructions required to process the data (and therefore increase the algorithm speed and reduce the power requirement).

The future work is to determine the exact compression scheme or analysis to perform on the tool holder, and then select a processor that can facilitate the timely execution of that algorithm with the least power consumption. Once selected, a new design could be created around the processor, and the actual program could be written.

5.2.4 Add a Charging Temperature Sensor

The current battery charging system does not contain a battery temperature sensor.

The temperature sensor provides an extra factor of safety when charging, as the ability to detect excess temperatures would allow the charging system to cease operation before the situation becomes hazardous. The lack of a temperature sensor is acceptable for prototypes designed for use in controlled research environments, but unacceptable for use in a commercial product. The Texas Instruments BQ2057 charging chip used in the current design already supports a temperature input, so all that would be required in future designs is a temperature sensor attached to the battery.

Future work would be to select an appropriate temperature sensor and integrate it physically with the battery. Only minor effort would be required to integrate it with the charging system. Alternatively, batteries that already have integrated temperature sensors could be used.

5.2.5 Reduce Strain Gauge Temperature Sensitivity

The DC offsets caused by temperature gradients across the strain gauge bridge can hinder the accurate measurement of average torque. One possible solution is to locate the strain gauges in a location where the temperature is more evenly distributed. A finite element analysis of the tool for temperature during cutting could provide suitable locations for gauge placement. Also, more sensitive gauges could be placed far from the cutting interface on the tool holder body, where any temperature gradient would be much smaller. The drawback of using more sensitive gauges is that they tend to be more sensitive to temperature, meaning they will pick up more stray temperature variation. Another option is to examine techniques for determining the cutting coefficients from the higher frequency components instead of the average, eliminating the need for DC

measurements.

5.2.6 Information Sharing Between Research Groups

Currently, there is no standardized format for the exchange of machining sensor data. This means that sharing of information between research groups is difficult at best. The XML schema developed for experiment archival (see Section 3.2) only partially solves the issue. While the sensor data can be easily interpreted, the experimental configuration sections are still ambiguous. Because the file's configuration section is determined by each research group according to its needs, it's possible (and likely) that each group will chose slightly different XML tag names for the same physical quantity. These name variations cause a problem when using automated tools on another research group's data. Presumably, all of the necessary information is contained in some form in the file, as otherwise information sharing is not possible. The information in the configuration sections must be transformed from one group's configuration format into another group's configuration format. Extensible Stylesheet Language Transformations (XSLT) [W3C XSLT] may provide a mechanism to solve this problem. XSLT is a language for transforming general XML from one layout to another.

Further investigation is required to determine if XSLT is a viable solution to the problem of information sharing, and, if so, to create a basic framework for converting one research group's format into another's.

5.2.7 eXist XML Database

Organizing machining data is another problem beyond simple archiving. If the data is to be useful in the future, not only must it have the experimental setup (as

discussed in Section 3.1), but it must also be in a location accessible to the researcher. Storing the data on a computer file system with hierarchical directories is the simplest approach, but it can be troublesome to search for specific experimental conditions without knowing exactly where in the directory structure such tests are located. Manual indexing schemes are possible, but prone to error. Using native XML databases to house the machining data is one promising option.

Native XML databases store files in such a way that the data can quickly be retrieved by searching for a portion or portions of the file. While native XML databases are not as mature as SQL databases, they are becoming more widespread for data already in an XML form. They show great promise for situations where the search terms of interest aren't known at the time of database creation, as they can search through the entire contents of the XML files, something that conventional databases can't easily perform.

Preliminary work was performed using the eXist XML Database [eXist] to store the XML files collected from the output of the UNH Smart Machining System Platform. Basic functionality has already been added to the XML Measurement MATLAB Toolbox discussed in Section 3.3. Rudimentary testing of the system indicated that the database usage was feasible. Part of the further work will need to be automating the process of adding newly collected data sets into the database. Additionally, the common database queries should be optimized through careful attention to the database query language to reduce their processing time.

5.2.8 Real-Time Controller for Chatter Suppression

While the chatter simulator developed in Section 4.3 was designed to evaluate chatter controllers for use with the Smart Tool Holder, no controller was actually developed during the course of this research. Future work should involve the creation of one or more chatter control strategies, and their testing on the chatter simulator.

One simple chatter controller is the one described in [Ismail and Ziaei 2002], where the spindle speed is ramped continuously up and down during chatter conditions and the spindle speed is held constant when there is no chatter. In this way the system does not alter its conditions unless there is a need. The controller would need to be altered to use a torque signal instead of a microphone signal.

Another possible control strategy is to estimate the chatter frequency from the torque signal and to move the spindle speed to an integer divisor of that frequency, where chatter cannot occur. This approach is complicated by the fact that the chatter frequency does not typically appear above the noise floor in stable cutting conditions. Furthermore, the tooth passing signal and tool runout can hamper the detection and isolation of the chatter frequency.

Once a chatter control strategy has been tested on the Simulink model and has been shown to behave appropriately, it can then be implemented in software and evaluated during actual cutting.

5.2.9 Extension of Chatter Simulator

The chatter simulator developed in Section 4.3 for the evaluation of chatter controllers has several limitations. The most critical limitation is that negative chip

thicknesses can be computed for large displacements of the cutting tool, leading to negative forces. In reality, this “jumping out of the cut” would produce zero force. Redesigning the simulator to check for negative chip thickness would help to improve the accuracy of the simulator for large displacements, though it would probably negatively impact the simulation speed. The future work is to investigate the possibility of changing the model to access the chip thickness as signal in the simulation. If accomplished, the signal could be manipulated to prevent it from taking on values below zero.

An additional extension to the chatter simulator would be to support unevenly spaced teeth on the cutting tool. Currently, the simulator uses the fact that the teeth are evenly spaced and stores the displacement of the last tooth rather than keeping track of the workpiece surface. The future work is to modify the simulator to track the workpiece surface instead of the displacement. Tracking the workpiece surface could also open up other interesting avenues of investigation, such as estimating surface finish for different chatter control strategies.

LIST OF REFERENCES

- Abele, E., and Fiedler, U., 2004. "Creating Stability Lobe Diagrams during Milling", *Annals of the CIRP*, 53/1/2004, pp. 309-312.
- Altintas, Y. 2000. *Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design*. Cambridge University Press, New York. ISBN: 0-521-65973-6
- Bluetooth, Bluetooth Special Interest Group, <http://www.bluetooth.org/>
- Cui, Y., 2008. "Tool Wear Monitoring for Milling by Tracking Cutting Force Model Coefficients", Masters Thesis, University of New Hampshire.
- Delio, T.S., United States Patent Number 5,170,358, "Method for Controlling Chatter in a Machine Tool", December 8, 1992.
- Desfosses, B., 2007. "An Improved Power Threshold Method for Estimating Tool Wear During Milling", Masters Thesis, University of New Hampshire.
- Drozda, T., Wick, C., Benedict, J.T., Veilleux, R.F., Bakerjian, R. 1993. *Tool and Manufacturing Engineers Handbook: a Reference Book For Manufacturing Engineers, Managers, and Technicians*. 4th ed. Society of Manufacturing Engineers.
- eXist, "Open Source Native XML Database" <http://www.exist-db.org/>
- Ismail F., Ziaei, R., 2002. "Chatter suppression in five-axis machining of flexible parts", *International Journal of Machine Tools and Manufacture*, Volume 42, Issue 1, pp. 115-122.
- Javorek, B., 2008 "Calibration of a Milling Force Model Using Feed and Spindle Power Sensors", Masters Thesis, University of New Hampshire.
- Jerard, R.B., Fussell, B.K., Xu, M., and Yalcin, C. 2006. "Process Simulation and Feedrate Selection for Three-axis Sculptured Surface Machining", *International Journal of Manufacturing Research*, 1(2), pp. 136-156.
- Jerard, R.B., and Ryou, O., 2006. "NCML: A Data Exchange Format for Internet-Based

- Machining”, *Int. J. Computer Applications in Technology*, Vol. 26, Nos. 1/2.
- JSR-275, Java Community Process Program, <http://jcp.org/en/jsr/detail?id=275>
- MTConnect, <http://www.mtconnect.org/>
- Nordic Semiconductor, <http://www.nordicsemi.com/>
- Ryou, O., 2001. “E-Commerce for the Metal Removal Industry”, Systems PhD Dissertation, University of New Hampshire.
- Ryou, O., and Jerard, R.B., 2001. “NCML: An Internet Compatible Data Exchange Format for Custom Machined Parts”, *Proceedings of the 2001 NSF Design, Manufacturing & Industrial Innovation Research Conference*, Jan 7-10, Tampa, Florida.
- Schmitz, T.L., Smith, K.S., 2009. *Machining Dynamics: Frequency Response to Improved Productivity*. Springer Science, New York. ISBN: 978-0-387-09644-5
- Schuyler, C.K., 2005. “Feedrate Optimization and Tool Condition Monitoring of Flat End Milling Operations Utilizing Spindle Motor Power”, Masters Thesis, University of New Hampshire.
- Suprock, C.A., Hassan, R.Z., Jerard, R.B., Fussell, B.K., 2008. “Predicting Endmill Tool Chatter with a Wireless Tool Tip Vibration Sensor”, *11th CIRP Conference on Modeling of Machining Operations*, Gaithersburg, Maryland.
- Suprock, C.A., Jerard, R.B., Fussell, B.K., 2009. “*In Situ* Chatter Frequency Prediction Using Torque Data from a Wireless Sensor Integrated Tool Holder”. Under Review ASME MSEC2009.
- W3C XML, “Extensible Markup Language (XML)”, <http://www.w3.org/TR/xml/>
- W3C XML Schema, <http://www.w3.org/XML/Schema>
- W3C XSLT, “XSL Transformations (XSLT)”, <http://www.w3.org/TR/xslt>
- Wang, H., Azuaje, F., Jung, B., Black, N., 2003. “A markup language for electrocardiogram data acquisition and analysis (ecgML)” *BMC Medical Informatics and Decision Making*, 3:4
- WellLogML: An XML Standard for Web-Based Exchange of Well Log Data, 1999. Petrotechnical Open Software Corp. <http://posc.org/ebiz/WellLogML/V1.0/Introduction.html>

Xerces-C++, The Apache Software Foundation, <http://xerces.apache.org/xerces-c/>

Xu, M., 2007. "Smart Machining System Platform for CNC Milling with the Integration of a Power Sensor and Cutting Model", Systems PhD Dissertation, University of New Hampshire.

APPENDIX A

SMART TOOL HOLDER MAIN BOARD LAYOUT, SCHEMATIC AND PARTS

The Smart Tool Holder main board contains the Atmel AVR microcontroller, RN-41 Bluetooth module, ADC, DAC, instrumentation amplifier, bar graph LEDs and various support hardware. The board has three sets of pin headers to connect the rest of the design. The six-pin header should be wired to the connector in the core of the tool holder that mates with the removable cutting tool's connector. The two-pin header should be connected to a user accessible switch to control power. The three-pin header should be connected to both the battery and to the charging board described in Appendix B. The electrical schematic should be consulted for details about the ordering of the individual pins in the headers.

Figures A.1 through A.6 show the layout of the printed circuit board. The board's dimensions are 1.2" by 1.9". All of these figures are shown with 2:1 scale.

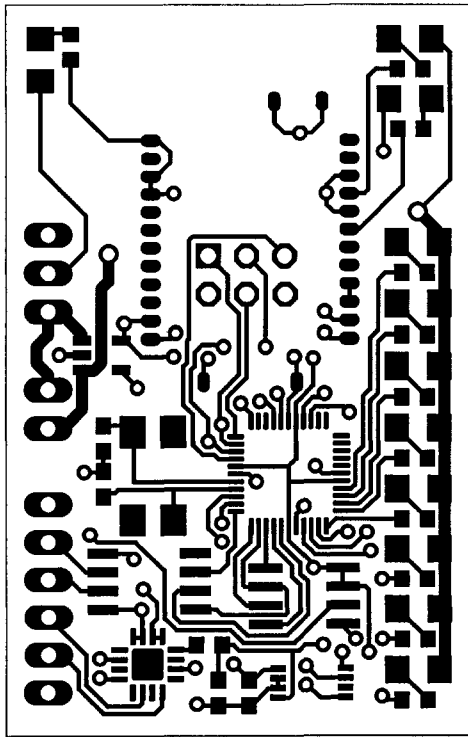


Figure A.1: Top copper

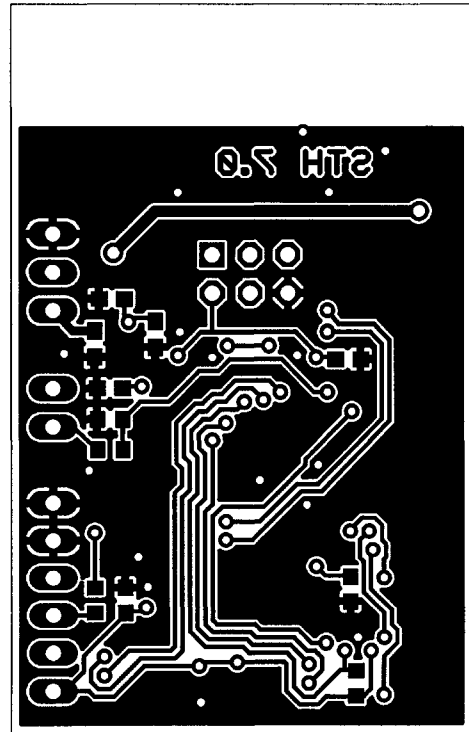


Figure A.2: Bottom copper

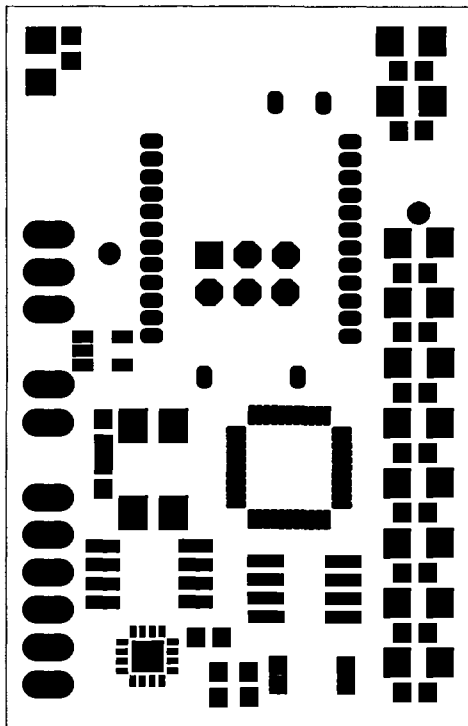


Figure A.3: Top solder mask

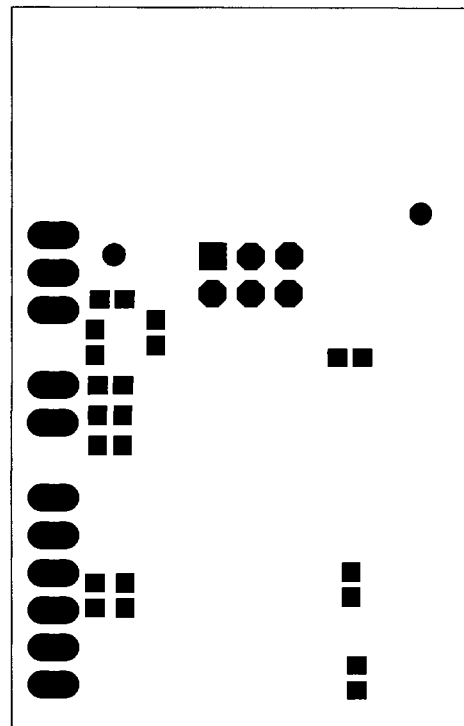


Figure A.4: Bottom solder mask

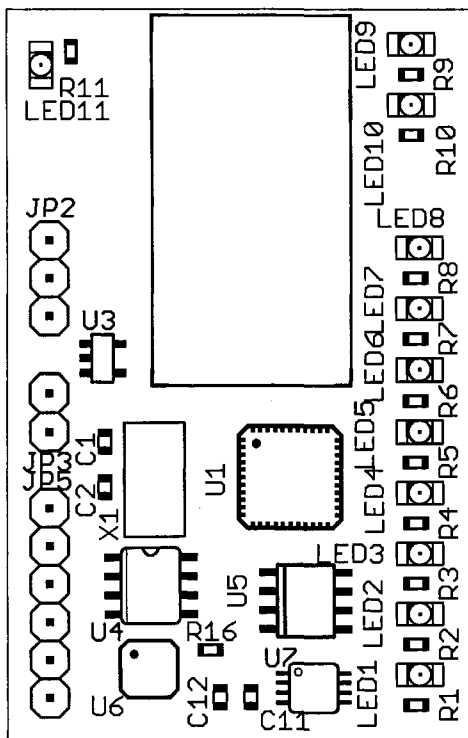


Figure A.5: Top silkscreen

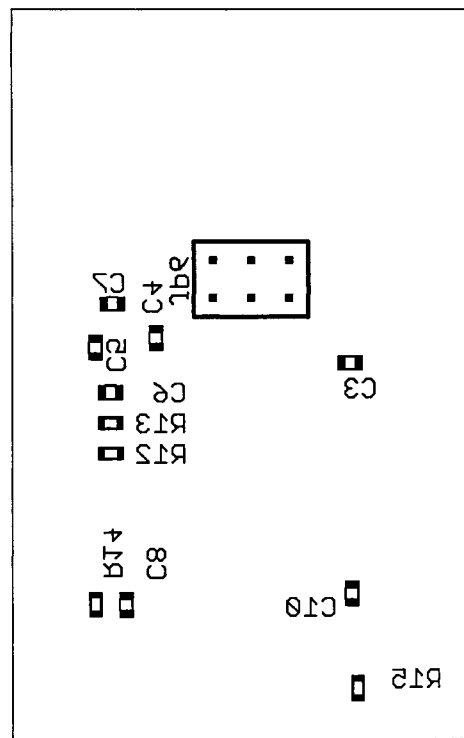


Figure A.6: Bottom silkscreen

Table A.1 gives a list of the parts contained on the Smart Tool Holder main electronics board. The references correspond to the component names on the electrical schematic.

Part Description	Qty.	References	Digi-key Catalog No.	Unit Price
22pF 0603 Capacitor	2	C1, C2	399-1053-1-ND	\$0.036
0.1uF 0603 Capacitor	5	C3, C7, C8, C10, C11	399-1097-1-ND	\$0.083
1uF 0603 Capacitor	1	C4	399-3118-1-ND	\$0.044
10uF Tantalum Capacitor	1	C5	511-1447-1-ND	\$0.36
0.01uF 0603 Capacitor	2	C6, C12	399-1092-1-ND	\$0.033
2-pin header	1	JP3	WM8096-ND	\$0.23
3-pin header	1	JP2	WM8097-ND	\$0.34
6-pin header	1	JP5	WM8100-ND	\$0.66
Green 1206 LED	4	LED1, LED2, LED3, LED10	754-1108-1-ND	\$0.52
Yellow 1206 LED	3	LED4, LED5, LED6	754-1114-1-ND	\$0.18
Red 1206 LED	3	LED7, LED8, LED11	754-1112-1-ND	\$0.18
Blue 1206 LED	1	LED9	754-1109-1-ND	\$0.52
220 ohm 0603 Resistor	11	R1- R11	RGH16P220CT-ND	\$0.324
100k ohm 0603 Resistor	4	R12, R13, R14, R15	RMCF1/16100K1%R CT-ND	\$0.16
5.6k ohm 0603 Resistor*	1	R16	P5.60KHCT-ND	\$0.073
AVR ATmega164P MLF44 Microcontroller**	1	U1	ATMEGA164P- 20MU-ND	\$4.82
Roving Networks RN-41 Bluetooth Module	1	U2	***	\$35.00
TPS73131 SOT23-5 LDO	1	U3	296-20788-1-ND	\$1.29

Table A.1: Main electronics board components

Part Description	Qty.	References	Digi-key Catalog No.	Unit Price
MAX6675 SO8 Temp. to Digital	1	U4	****	\$7.18
AD7391 SO8 DAC	1	U5	AD7391ARZ-ND	\$7.30
AD8557 LFCSP16 In-Amp	1	U6	AD8557ACPZ-REEL7CT-ND	\$5.02
AD7680 MSOP8 ADC	1	U7	AD7680BRMZ-ND	\$11.20
7.3728MHz FQ7050B Crystal	1	X1	631-1028-1-ND	\$0.96

Table A.1: continued

* The value of R16 sets the corner frequency of the first order anti-alias filter. The following equation gives the corner frequency:

$$f_0 = \frac{1}{2\pi * R_{16} * C_{12}} = \frac{1}{2\pi * 5.6\text{kohm} * 10\text{nF}} = 2.84\text{kHz} . \quad (\text{A.1})$$

** To program the ATmega164P, an AVR hardware programmer is required. Examples of these include the Atmel STK500 (Digi-key part no. ATSTK500-ND) or the Atmel AVRISP mkII (Digi-key part no. ATAVRISP2-ND).

*** The RN-41 is available from Mouser, with part number 765-RN-41.

**** The MAX6675 is available from maxim-ic.com with part number MAX6675ISA+.

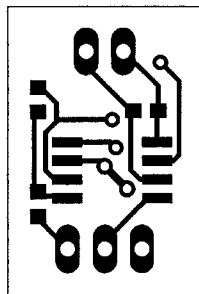
APPENDIX B

CHARGING BOARD LAYOUT, SCHEMATIC AND PARTS

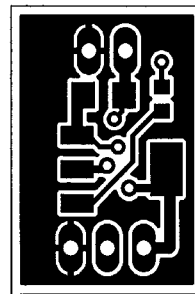
This board contains the circuitry necessary for charging a Lithium-polymer battery. The Texas Instruments BQ2057 charge management IC is utilized in the design to regulate the voltage and current going to the battery. A PNP transistor on the underside of the board is used by the BQ2057 to actually pass the charge to the battery, since the small package of the charging IC doesn't lend itself well to power dissipation. A 0.3 ohm current sense resistor is used by the BQ2057 to detect the charging current and limit it to a safe level. This circuit was modeled after the typical application circuits found in the BQ2057 data sheet.

The charging board has two sets of pin headers: a two-pin and a three-pin. The two-pin header at the top of the board is used to connect to the charging jack. The polarity of the header is marked on the silkscreen layer. A regulated 5V power supply capable of supplying at least 0.3A should be used to charge the system. The three-pin header at the bottom of the board should be connected to both the battery and to the STH7 main board. The three pins connect to the three pin header on the main board, but the positive and negative pins must be additionally connected to the battery to provide power to the system. This design was chosen because it provided space savings on the main board instead of having an additional pin header for the battery. The middle pin is used to drive a LED on the main board to indicate when the battery is being charged. The polarity of the other two pins are marked on the silkscreen layer.

The following figures show the layout of the PCB. The board's dimensions are 0.5" by 0.75". All figures are shown with 2:1 scale.



*Figure B.1:
Top copper*



*Figure B.2:
Bottom
copper*

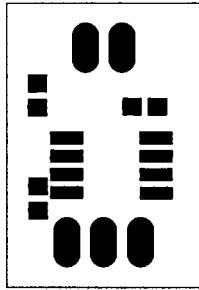


Figure B.3:
Top solder
mask

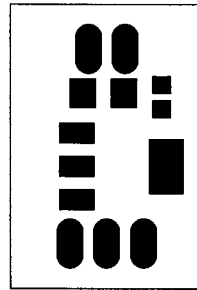


Figure B.4:
Bottom solder
mask

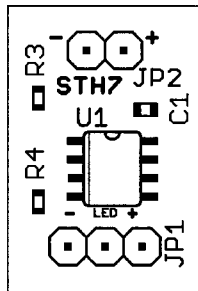


Figure B.5:
Top silkscreen

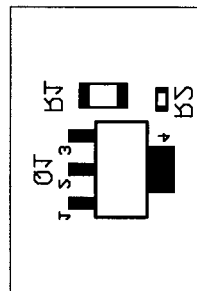


Figure B.6:
Bottom
silkscreen

Table B.1 gives a list of the parts contained on the charging board. The references correspond to the component names on the electrical schematic.

Part Description	Qty.	References	Digi-key Catalog No.	Unit Price
0.1uF 0603 Capacitor	1	C1	399-1097-1-ND	\$0.083
3 pin right angle 0.1" male header	1*	JP1	WM8097-ND	\$0.34
2 pin right angle 0.1" male header	1*	JP2	WM8096-ND	\$0.23
FZT788B SOT223 Transistor	1	Q1	FZT788BCT-ND	\$1.47
0.3 ohm 1206 Current Sense Res.	1	R1	CSR1/20.31%ICT-ND	\$0.69
1k ohm 0603 Resistor	1	R2	P1.00KHCT-ND	\$0.073
5.6k ohm 0603 Resistor	1	R3	P5.60KHCT-ND	\$0.073
4.7k ohm 0603 Resistor	1	R4	P4.70KHCT-ND	\$0.073
TI BQ2057 SO8 Li-poly charger	1	U1	296-9360-5-ND	\$1.68

Table B.1: Charging board components

* JP1 and JP2 are optional. The connecting wires can be soldered directly to the board instead of connecting to the pin headers. This can be advantageous when space is at a premium.

APPENDIX C

ADDITIONAL ELECTRONICS PARTS

Several additional parts are required beyond the components listed in Appendix A and Appendix B. These components are summarized in Table C.1. Connecting wire is not listed, but any insulated wire in the range of 24-30 AWG will suffice.

Part Description	Qty.	Supplier	Part No.	Unit Price
Strain gauge bridge	1	Omega	SGT-2DD/350-SY41	\$10.60
Sealed power switch	1	Digi-key	360-1774-ND	\$5.20
2 pin connector housing	2*	Digi-key	WM2800-ND	\$0.55
3 pin connector housing	2*	Digi-key	WM2801-ND	\$0.71
6 pin connector housing	1	Digi-key	WM2804-ND	\$0.82
24-30 AWG crimp connectors**	16***	Digi-key	WM2562CT-ND	\$0.0491
3.7V 430mAh Lithium-Polymer battery	1	Max Amps	Lipo-430-loose-cell	\$7.99
2.5 mm DC charging jack	1	All Electronics	DCJ-6	\$0.20
5V 3.7A DC charger	1	All Electronics	PS-537	\$4.75

Table C.1: Additional Smart Tool Holder parts

* One of each of the 2 and 3 pin connector housings are optional, as per the notes on JP1 and JP2 in Appendix B.

** The crimp connectors can be crimped using a Molex Universal Crimp Tool (Digi-key part no. WM9999-ND).

*** Though only 16 are required for the design, as many as 50% extra additional crimps should be allocated for attrition purposes.

APPENDIX D

XML SCHEMA

The schema for the experiment results file type is given here:

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:exp="http://www.unh.edu/dml"
  targetNamespace="http://www.unh.edu/dml"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <annotation>
    <documentation xml:lang="en">
      Experiment result schema for DML
      Copyright 2007 Design and Manufacturing Laboratory
    </documentation>
  </annotation>

  <element name="experimentResults" type="exp:ExperimentType" />
  <element name="interface" type="exp:InterfaceType" />
  <element name="sensor" type="exp:SensorType" />
  <element name="personnel" type="exp:PersonnelType" />
  <element name="configuration" type="exp:ConfigurationType" />

  <complexType name="ExperimentType">
    <sequence>
      <element ref="exp:personnel" />
      <element ref="exp:configuration" />
      <element name="measurement" type="exp:MeasurementType"
maxOccurs="unbounded" />
    </sequence>
  </complexType>

  <complexType name="PersonnelType">
    <sequence>
      <element name="experimenter" type="string"
maxOccurs="unbounded" />
    </sequence>
  </complexType>

  <complexType name="MeasurementType">
    <sequence>
      <element name="startState" type="exp:StateType" />
      <element name="endState" type="exp:StateType" minOccurs="0" />
      <element ref="exp:interface" minOccurs="1"
maxOccurs="unbounded" />
    </sequence>
```

```

    <attribute name="start" type="dateTime" use="required" />
    <attribute name="end" type="dateTime" use="optional" />
</complexType>

<complexType name="InterfaceType">
  <sequence>
    <element name="attribute" type="exp:Attribute" minOccurs="0"
maxOccurs="unbounded" />
    <element name="sensor" type="exp:SensorType" minOccurs="1"
maxOccurs="unbounded" />
  </sequence>
  <attributeGroup ref="exp:NameAndType" />
</complexType>

<complexType name="SensorType">
  <sequence>
    <element name="attribute" type="exp:Attribute" minOccurs="0"
maxOccurs="unbounded" />
    <element name="channel" minOccurs="0" maxOccurs="unbounded">
      <complexType>
        <sequence>
          <element name="attribute" type="exp:Attribute"
minOccurs="0" maxOccurs="unbounded" />
          <choice>
            <element name="dataGroup" type="exp:DataGroupType"
maxOccurs="1" />
            <element name="data" type="exp:DataType"
maxOccurs="unbounded" />
          </choice>
        </sequence>
        <attribute name="name" type="string" use="required" />
        <attribute name="units" type="string" use="required" />
      </complexType>
    </element>
  </sequence>
  <attributeGroup ref="exp:NameAndType" />
</complexType>

<attributeGroup name="NameAndType">
  <attribute name="name" type="string" use="required" />
  <attribute name="type" type="string" use="required" />
</attributeGroup>

<complexType name="Attribute">
  <attribute name="setting" type="string" use="required" />
  <attribute name="value" type="string" use="required" />
</complexType>

<complexType name="DataGroupType">
  <simpleContent>
    <extension base="string">
      <attribute name="samplefreq" type="decimal" use="required" />
      <attribute name="encoding" default="whitespace">
        <simpleType>

```

```

        <restriction base="string">
            <enumeration value="whitespace" />
            <enumeration value="comma" />
            <enumeration value="tab" />
            <enumeration value="base64" />
        </restriction>
    </simpleType>
</attribute>
</extension>
</simpleContent>
</complexType>

<complexType name="DataType">
    <simpleContent>
        <extension base="decimal">
            <attribute name="time" type="decimal" use="required" />
        </extension>
    </simpleContent>
</complexType>

<complexType name="ConfigurationType">
    <sequence>
        <any processContents="lax" minOccurs="0" maxOccurs="unbounded" />
    </sequence>
</complexType>

<complexType name="StateType">
    <sequence>
        <any processContents="lax" minOccurs="0" maxOccurs="unbounded" />
    </sequence>
</complexType>
</schema>

```

APPENDIX E

EXAMPLE XML DATA FILE

The following XML is a example experimental results file. The bulk of the sample points in each <dataGroup> have been removed to fit the document within a reasonable number of pages.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<experimentResults xmlns="http://www.unh.edu/dml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.unh.edu/dml experiment.xsd">

  <personnel>
    <experimenter>Yanjun</experimenter>
  </personnel>

  <configuration>
    <version>Testing 0.6</version>
    <purpose>Wear test
half immersion upmill
Does the upmill wear more quickly than the downmill?</purpose>
    <machine>
      <manufacturer>Fadal</manufacturer>
      <model>EMC</model>
      <serial-number>012004107001</serial-number>
      <type>3-Axis Vertical Mill</type>
    </machine>
    <runout units="Inches">0.000</runout>
    <workpiece>
      <material>1018 Steel</material>
      <dimensions>8x6x1</dimensions>
    </workpiece>
  </configuration>

  <measurement end="2007-07-30T11:19:02" start="2007-07-30T11:19:01">
    <startState>
      <line-number>26</line-number>
      <tare-power units="Horsepower">2257.2</tare-power>
      <spindle-speed units="RPM">2500</spindle-speed>
      <feed-rate units="Inches/Min">5.883048</feed-rate>
      <xpos units="Inches">1.387301</xpos>
    </startState>
  </measurement>
</experimentResults>
```

```

    <ypos units="Inches">0.000000</ypos>
    <zpos units="Inches">-0.150000</zpos>
    <average-contact-area units="Inches^2">0.058903</average-contact-
area>
    <instantaneous-contact-area
units="Inches^2">0.058903</instantaneous-contact-area>
    <average-material-removal-rate
units="Inches^3/Min">0.231779</average-material-removal-rate>
    <axial-depth units="Inches">0.149995</axial-depth>
    <radial-depth units="Inches">0.249999</radial-depth>
    <FRA-length-of-move units="Inches">0.850000</FRA-length-of-move>
    <actual-length-of-move units="Inches">0.850000</actual-length-of-
move>
    <actual-move-time units="Seconds">0.144312</actual-move-time>
    <tool>
      <name>T1</name>
      <material>Carbide</material>
      <type>1</type>
      <flutes>1</flutes>
      <helix-angle units="Degrees">30.000000</helix-angle>
      <diameter units="Inches">0.500000</diameter>
      <flute-length units="Inches">2.500000</flute-length>
      <part-number>1/2-EM</part-number>
    </tool>
  </startState>
  <endState>
    <xpos units="Inches">1.505919</xpos>
    <ypos units="Inches">0.000000</ypos>
    <zpos units="Inches">-0.150000</zpos>
  </endState>

  <interface name="ATD 0" type="cb atd">
    <attribute setting="BoardNumber" value="0"/>
    <attribute setting="SampleFreq" value="5000"/>
    <sensor name="Load Cell" type="Kistler">
      <channel name="Fx" units="Pounds">
        <attribute setting="DataToUnitConversion" value="linear"/>
        <attribute setting="UnitsPerData" value="112.400002"/>
        <attribute setting="UnitsAtDataZero" value="0.770000"/>
        <attribute setting="ATDChannelNumber" value="0"/>
        <dataGroup encoding="whitespace" samplefreq="5000">0.0265503
0.0170898 (removed 5998 samples for example) -0.00488281 </dataGroup>
      </channel>
      <channel name="Fy" units="Pounds">
        <attribute setting="DataToUnitConversion" value="linear"/>
        <attribute setting="UnitsPerData" value="-112.400002"/>
        <attribute setting="UnitsAtDataZero" value="-0.300000"/>
        <attribute setting="ATDChannelNumber" value="1"/>
        <dataGroup encoding="whitespace" samplefreq="5000">0.00732422
0.00274658 (removed 5998 samples for example) 0.0259399 </dataGroup>
      </channel>
      <channel name="Fz" units="Pounds">
        <attribute setting="DataToUnitConversion" value="linear"/>
        <attribute setting="UnitsPerData" value="112.400002"/>

```



```

        <attribute setting="UnitsAtDataZero" value="0.940000"/>
        <attribute setting="ATDChannelNumber" value="2"/>
        <dataGroup encoding="whitespace" samplefreq="5000">0.0131226
-0.038147 (removed 5998 samples for example) -0.0201416 </dataGroup>
    </channel>
</sensor>
<sensor name="Remaining" type="Composite">
    <channel name="PowerL" units="Horsepower">
        <attribute setting="DataToUnitConversion" value="linear"/>
        <attribute setting="UnitsPerData" value="0.200000"/>
        <attribute setting="UnitsAtDataZero" value="0.000000"/>
        <attribute setting="ATDChannelNumber" value="3"/>
        <dataGroup encoding="whitespace" samplefreq="5000">2.81464
2.78748 (removed 5998 samples for example) 2.77527 </dataGroup>
    </channel>
    <channel name="Position" units="Volts">
        <attribute setting="DataToUnitConversion" value="linear"/>
        <attribute setting="UnitsPerData" value="1.000000"/>
        <attribute setting="UnitsAtDataZero" value="0.000000"/>
        <attribute setting="ATDChannelNumber" value="4"/>
        <dataGroup encoding="whitespace" samplefreq="5000">5.10895
4.92126 (removed 5998 samples for example) 4.4519 </dataGroup>
    </channel>
    <channel name="Accel" units="Volts">
        <attribute setting="DataToUnitConversion" value="linear"/>
        <attribute setting="UnitsPerData" value="1.000000"/>
        <attribute setting="UnitsAtDataZero" value="0.000000"/>
        <attribute setting="ATDChannelNumber" value="5"/>
        <dataGroup encoding="whitespace" samplefreq="5000">0.177002
0.0848389 (removed 5998 samples for example) -0.541382 </dataGroup>
    </channel>
</sensor>
</interface>
</measurement>
</experimentResults>

```

APPENDIX F

DESCRIPTION OF MATLAB TOOLBOX

Once the XML Measurement Toolbox has been installed on a computer, the tools can be accessed from MATLAB's Start button, located in the lower left hand corner of the MATLAB desktop (see Figure F.1). Each of the tools can be run directly from the menu, or they can instead be accessed from the Main GUI (see Figure F.2).

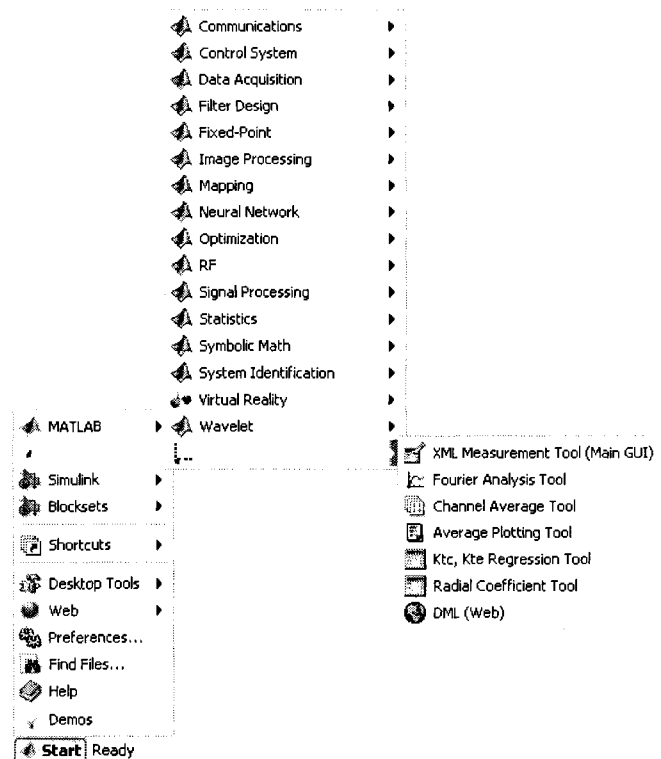


Figure F.1: XML Measurement Toolbox accessed from the MATLAB Start menu

When run directly from the menu, the tools will prompt the user to load channel data from XML files or from an eXist XML database. The Main GUI, however, allows a

user to load the data set once and use multiple tools to examine it. For this reason, the use of the Main GUI is generally preferred over running the tools directly.

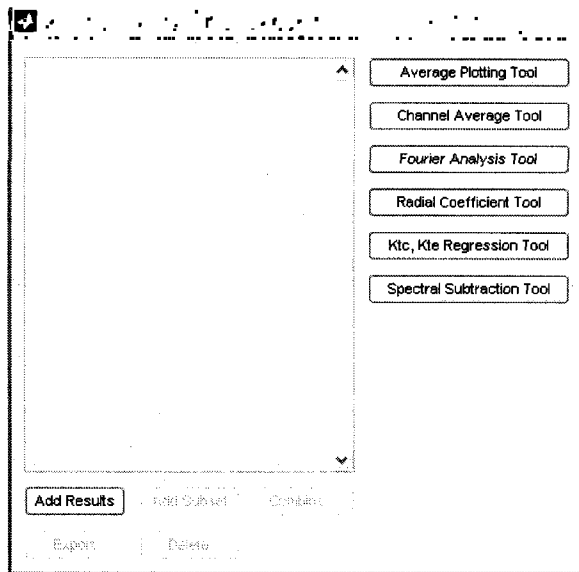


Figure F.2: Main GUI

The Main GUI has a central list box where the result sets are loaded and manipulated. A result set is simply a collection of channel data from one or more files. New result sets can be loaded from both XML files and eXist XML Databases. The list of buttons on the right hand side of the Main GUI correspond to the available analysis tools. An analysis tool can be run on any one of the result sets present in the central list box. Furthermore, the Main GUI offers the ability to create new result sets from either the combination of other result sets or from a subset of the channels contained in a single result set.

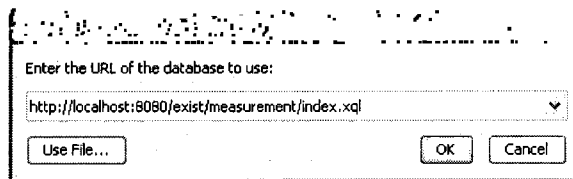


Figure F.3: XML database selection dialog

When loading result sets, the user is presented with the XML database selection dialog (see Figure F.3). From there, the user can enter the URL of an eXist XML database, or choose to load data from files (see Figure F.4). Once the desired channels have been loaded either from a database or from files, the Main GUI will prompt the user for a name for the result set, to distinguish it from other sets.

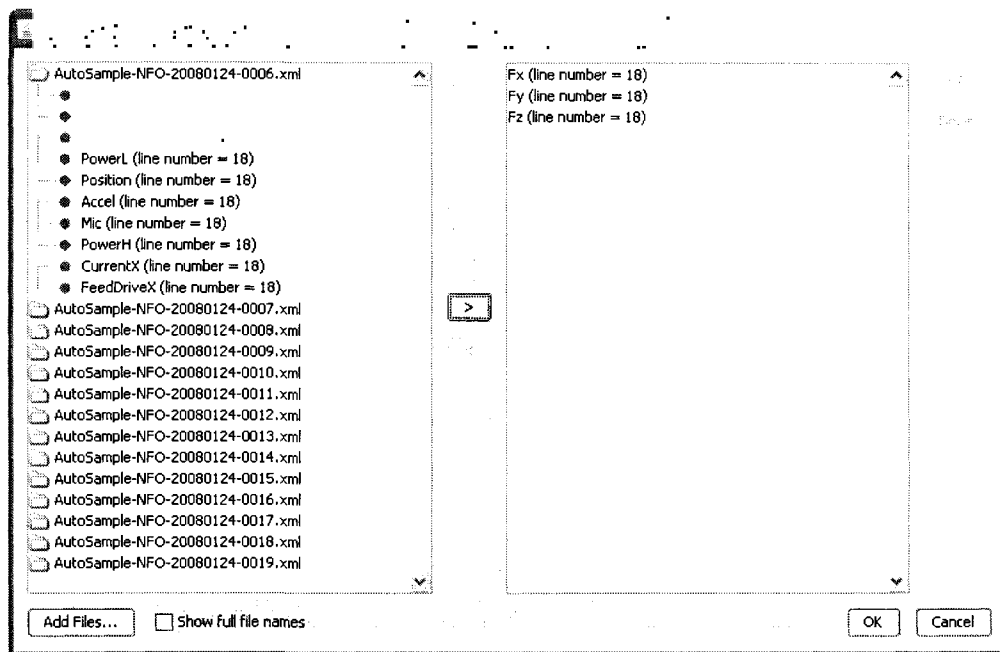


Figure F.4: Load from files dialog

From the Main GUI, one can create new result sets from the combination of two or more other sets through the use of the *Combine* button. The program will prompt the user for the desired sets to merge, and then ask for a name for the new set. One can also create a new result set from a subset of the channels contained in another set through the use of the *Add Subset* button (see Figure F.5).

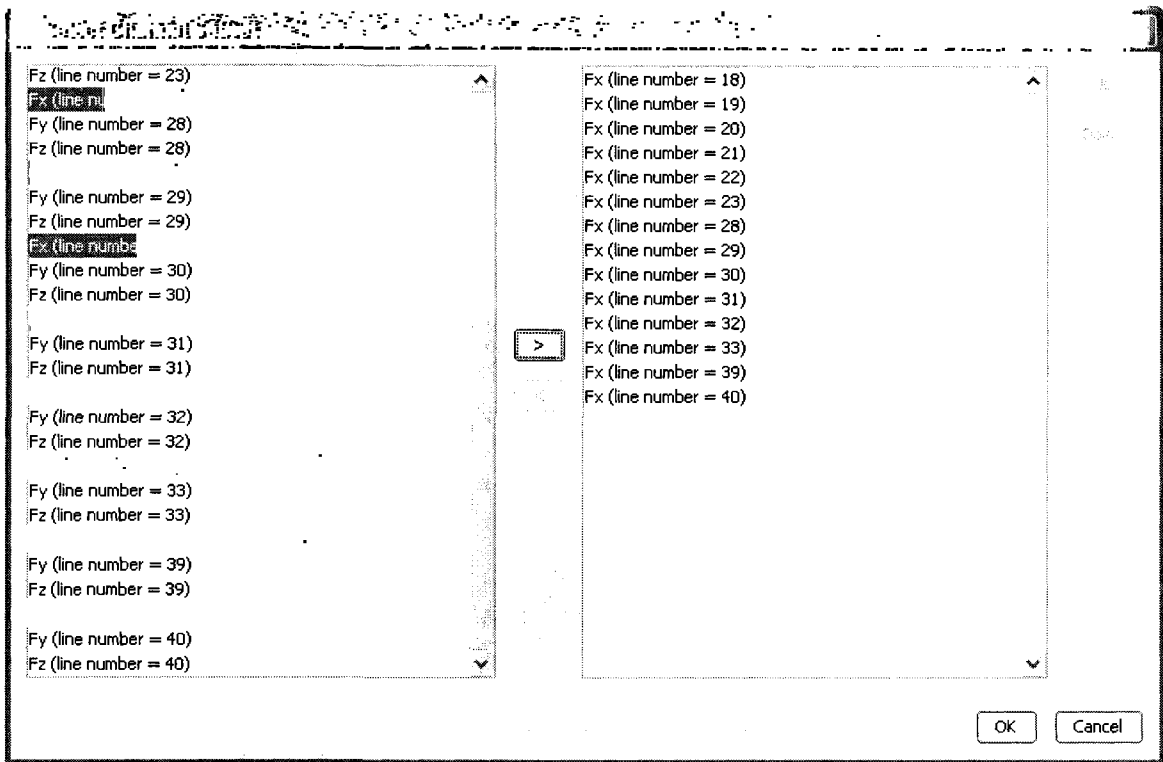


Figure F.5: Subset dialog box

The Main GUI also offers an export option to allow a user to access the result set data from the MATLAB workspace. The *Export* button prompts the user for a list of result sets to export, and their desired names in the MATLAB workspace (for cases where their current names are already in use by other variables). After the export process, the user can interact with the result set objects. The most common usage is to access the data and time vectors, which can be accomplished with the following commands:

```
channels = result.getChannelArray;
channell = channels(1);
[data, time] = getchanneldata(channell, []);
```

where *result* is the exported result set, and *data* and *time* are the desired vectors. Further examples of manipulating result sets can be found in the source code of the various

analysis tools, stored in the *toolbox/xmlmeasurement/tools/* directory. The MATLAB command line help is another good reference for information about the toolbox commands.

Repetitive tasks can be automated through the creation of a new tool. Any MATLAB “m” file located in the tools directory will automatically be included as part of the Main GUI's list of tools. The simplest method for creating a new tool is to copy a tool that already provides some of the basic functionality, and extend the tool from there.

APPENDIX G

FORCE AND TORQUE MODEL MATRIX GENERATION PROGRAM

This MATLAB program computes the **F** and **T** matrices described in Section 4.2.

```
function [F,T] = getcoefficients(phi, N, zsteps, a, K, alpha_helix, ...
    R, runout, phi_st, phi_ex)

% Written by Jeffrey Nichols
% Copyright 2009 Design and Manufacturing Laboratory

% This program takes uses 5-dimensional "matrices" to compute the
% 3-dimensional F and T matrices.
%
% The Cartesian forces at angle phi can be found using this pseudocode:
% [F_x(phi);F_y(phi);F_z(phi)] = F(:, :, phi) * [x;y;1]
%
% The torque at angle phi can be found using this pseudocode:
% Torque(phi) = T(:, :, phi) * [x;y;1]

% get the runout magnitude and angle
rho = abs(runout);
beta = angle(runout);

% set up z for integration
dz = a/zsteps;
z = 0:dz:a-dz;

% set up phi, z and j in 3-d
% their dimensions are [phi,z,j]
[phi,z,j] = ndgrid(phi,z,0:N-1);

% compute the angle for every point on the tooth.
% its dimension is [phi,z,j]
phi_j = phi + j*2*pi/N - z*tan(alpha_helix)/R;

% precompute the sines and cosines of phi_j, as they're used a lot.
% their dimension is [1,1,phi,z,j]
spj = reshape(sin(phi_j), [1,1,size(phi_j)]);
cpj = reshape(cos(phi_j), [1,1,size(phi_j)]);
```

```

% chip thickness change due to runout for every spot on the tooth.
% its dimension is [phi,z,j]
rho_j = rho*cos(beta - phi_j - phi);
% subtract the runout from the last tooth
rho_j = rho_j - cat(3,rho_j(:,:,end),rho_j(:,:,1:end-1));

% the chip thickness components.
% their dimension is [1,1,phi,z,j]
h_x = spj;
h_y = cpj;
h_c = reshape(rho_j,[1,1,size(rho_j)]);

% probably a better way to do this, but this works and makes the rest
% of the program a lot more readable
K_tc = K(1,1);
K_te = K(1,2);
K_rc = K(2,1);
K_re = K(2,2);
K_ac = K(3,1);
K_ae = K(3,2);

% the differential forces in cylindrical are:
% [dF_tx dF_ty dF_tz;dF_rx dF_ry dF_rz;dF_ax dF_ay dF_az]
% its dimension is [3,3,phi,z,j]
dF_cyl = [h_x*K_tc,h_y*K_tc,h_c*K_tc+K_te;...
          h_x*K_rc,h_y*K_rc,h_c*K_rc+K_re;...
          h_x*K_ac,h_y*K_ac,h_c*K_ac+K_ae];

% do [-cpj,-spj,0;spj,-cpj,0;0,0,1]*dF_cyl to get Cartesian.
% have to do it component by component since MATLAB only likes matrix
% multiplication on 2-d matrices.
% its dimension is [3,3,phi,z,j]
dF_cart = [-cpj.*dF_cyl(1,1,:,::)-spj.*dF_cyl(2,1,:,::),...
           -cpj.*dF_cyl(1,2,:,::)-spj.*dF_cyl(2,2,:,::),...
           -cpj.*dF_cyl(1,3,:,::)-spj.*dF_cyl(2,3,:,::);...
           spj.*dF_cyl(1,1,:,::)-cpj.*dF_cyl(2,1,:,::),...
           spj.*dF_cyl(1,2,:,::)-cpj.*dF_cyl(2,2,:,::),...
           spj.*dF_cyl(1,3,:,::)-cpj.*dF_cyl(2,3,:,::);...
           dF_cyl(3,1,:,::),...
           dF_cyl(3,2,:,::),...
           dF_cyl(3,3,:,::)];

% this function determines if the tooth is in the material
% its dimension is (currently) [1,1,phi,z,j]
g = (spj > 0) & (cos(phi_ex) <= cpj & cpj <= cos(phi_st));
% make its dimension [3,3,phi,z,j] by duplication
g = repmat(g,[3,3,1,1,1]);

```



```

% use the g matrix to zero the forces outside of the cut
dF_cart(~g) = 0;
dF_cyl(~g) = 0;

% sum over each tooth and integrate in z to get the final force matrix
% F has dimension [3,3,phi]
F = sum(sum(dF_cart,5),4)*dz;

% sum over each tooth and integrate in z to get torque from tangential
% force.
% T has dimension [1,3,phi]
T = R*sum(sum(dF_cyl(1, :, :, :, :),5),4)*dz;

```