Permanent-Magnet Synchronous Motors and Associated
Power Electronics for Direct-Drive Vehicle Propulsion

by

John Ofori-Tenkorang

S.B., Massachusetts Institute of Technology (1989)
S.M., Massachusetts Institute of Technology (1993)
E.E., Massachusetts Institute of Technology (1993)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Science

at the

MASSACHУСЕTTS INSTITУTE OF TECHNOLOGY

September 1996

©1996, Massachusetts Institute of Technology. All Rights Reserved.

Author ..........................................................

Department of Electrical Engineering and Computer Science
September 11th, 1996

Certified by .................................................

Jeffrey H. Lang
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .................................................

Frederic R. Morgenthaler
Chairman, Departmental Committee on Graduate Students
Permanent-Magnet Synchronous Motors and Associated Power Electronics for Direct-Drive Vehicle Propulsion

by

John Ofori-Tenkorang

Submitted to the Department of Electrical Engineering and Computer Science on September 11th, 1996, in partial fulfillment of the requirements for the degree of Doctor of Science

Abstract

The quest for improved air quality in urban areas and improved automobile fuel efficiency has rekindled interest in the development of all-electric and hybrid vehicles. The lack of an economical low-weight on-board energy storage medium with high power and energy density remains a major obstacle to the commercialization of electric vehicles. Problems with energy storage notwithstanding, the efficiency of the powertrain can be improved and its complexity reduced by employing a direct-drive wheel motor propulsion system. By integrating the drive motor with the wheel, one can eliminate transaxles, differentials, clutches and reduction gears, thus simplifying the mechanical design of the vehicle. With wheel motors, losses deriving from the mechanical transmission and the drive line are eliminated, thereby yielding significant improvements in the efficiency of the drive system from battery to wheel. This thesis develops such a drive system.

The availability of high-energy Neodymium-Iron-Boron magnets, coupled with the emergence of high-power, fast-switching, third generation IGBTs make a highly-efficient low-weight direct-drive powertrain possible. A 20 hp high-torque, low-weight, permanent-magnet synchronous wheel motor and the associated power electronics was designed, fabricated and experimentally evaluated. The motor fits in a 19-inch wheel and produces 450 Nm of peak torque from 0 to 325 rpm, with constant power operation up to a maximum speed of 1300 rpm. A 4:1 speed range is achieved with a bi-directional boost converter in the dc link of a 3-phase PWM voltage source inverter. The active mass of the resulting motor is 25.1 kg, and the total efficiency from traction power source to wheel is approximately 85% over the Federal Urban Drive Schedule and 92% over the Federal Highway Drive Schedule.

Finally this thesis answers the question of when to use a Halbach magnet array in the design of permanent-magnet synchronous motors. We conclude that for applications that preclude the use of a magnetic rotor backing, such as high-speed energy-storage flywheels, a Halbach array produces higher torque per volume of magnet material than a conventional array. With a permeable rotor backing, a conventional array produces higher torque per magnet volume in the regime of thin magnets. As the magnet thickness increases, a Halbach array offers superior performance.

Thesis Supervisor: Jeffrey H. Lang
Title: Professor of Electrical Engineering and Computer Science
Acknowledgments

Funding for this work was provided by Ford Motor Company under subcontract number 47-2-RN0095. I would like to thank Mr. Brad Bates and Mr. Allan Gale, both at Ford Motor Company, for funding this project and for the several valuable discussions.

At MIT, I would like to thank my thesis supervisor, Professor Jeff Lang, for his patience, guidance and support throughout this work. I have learned a lot from Jeff over the past 3 years, and working with him was great fun. I thank Professors Jim Kirtley, David Trumper and John Kassakian for serving as readers for this thesis.

Several people have been of great help during the experimental phase of this thesis. My thanks go to Wayne Ryan at MIT for machining the inverter parts and helping with the construction of the inverter and to Tracy Clark and Dave Otten for helping me decode some of the obscure inner workings of the MC68HC16Z1 microprocessor. Much thanks to Deron Jackson and Professor Steven Leeb for letting me borrow test equipment and ad hoc electronic parts when I couldn't afford to wait for a purchase order to go through. Thanks to Wayne Hagman an Vivian Mizuno for helping me repair my inverter when an IGBT decided to go up in smoke on a Friday night! Thanks to Carol Choi for helping with the drawing of circuit schematics.

I thank my colleagues and friends at LEES, Khurram Afridi, Dave Perreault, Mark Thompson, Brian Perreault, Robert Selders, Mary Tolikas, Kamakshi Srinivasan, Jeff Chapman, Ahmed Mitwalli, Julio Castrillon-Candas, Vahe Caliskan, Jama Mchammed, Jesko Hagee, and ... (the list goes on) for making this Lab a fun place study.

The prototype wheel motor presented in this thesis was built and tested at the Superior Electric Company (SECO) in Bristol, CT. Several people at SECO have been of tremendous help for which I am very grateful. My first thanks go to Dr. Marty Kaplan, Vice President of Engineering at SECO, for supervising the motor construction project. I thank Dr. Peter Senak designing the apparatus needed to wind the motor and providing me with all the necessary support needed to wind the motor and get it running. Thanks to Tom Corlozzi for setting up the Powerstat and rectifier set that was used as the dc power source to the inverter, and to Willis Dudley for his help with setting up the thermocouple readers. I thank Dick Connors for designing the mechanical parts needed to assemble the motor and Lee Faucher for machining all the parts and assembling the motor. Thanks to Joe Levesque, Ron Strid and Art Johnson for helping with mounting of the motor on the dynamometer. Special thanks to Jim Burns for setting up the dynamometer and bringing me up to speed on how to operate the dyno. Thanks to Lane Woodland for all his help including lending me a laptop so that I could edit my thesis in my hotel room while watching Atlanta '96 on CNN. Thanks to Mark Rollins, Charlie Bald, and Mark Beaulieu for all their help and making me feel at home at SECO.

Back to MIT where I have spent 11 years (whew!) as a student, I would like to thank some very special people. If I were asked to name a mentor during my tenure at MIT, it would be Professor John G. Kassakian. I first met John during my first semester, sophomore year at MIT when he taught me 6.002 (Circuits and Electronics, for the non-MIT types).
John has always believed in me, given me great advice and most importantly opportunity, for which I am extremely grateful for. Other professors who have also taken special interest in my education, professional growth and personal well-being at MIT are Professors Markus Zahn, Leonard Gould and Robert Kingston (my undergraduate academic advisor). To them I extend my profound gratitude.

Like they say ... all work and no play ..., you know what I mean! Eleven years in Boston wouldn’t have been any fun without great friends like Albert Essiam, Mike Owu, Mawuli Tse, Evelyn Anim, Gillian Brown, Madeline Jeune, Elizabeth N. Ngonzi, Khaitsa Wasiyo, Kofi Fynn, Kamel Addo, the late John A. Selormey ’97 and many others whose names I haven’t mentioned. A big thanks to all my African brothers and sisters in the Africans Association at MIT and African-American brothers and sisters in the BGSA for all the good times we have shared.

Finally I would like to thank Dr. & Mrs. Clarence G. Williams for being my parents away from home.
To my mother and my father,
for the huge sacrifices they made,
so that I may get an education!
# Contents

1 Introduction ................................................. 16

1.1 Background .................................................. 16

1.2 Technological Challenges ................................. 19

1.2.1 Energy Storage ............................................. 19

1.2.2 Current EV Powertrains ................................. 21

1.3 Towards a more Efficient Electric Powertrain .......... 22

1.4 Enabling Technologies ...................................... 25

1.4.1 Permanent Magnets ...................................... 25

1.4.2 Power Semiconductor Switches ......................... 26

1.5 Thesis Objectives and Contributions .................... 28

1.6 Thesis Organization .......................................... 29

2 Torque Production in Slotless Halbach and Conventional Permanent-Magnet Motors: A Comparative Analysis 31

2.1 Introduction .................................................. 31

2.2 The Halbach Array: An Intuitive Understanding of Spatial Flux Distribution 35

2.3 Torque Computation .......................................... 37

2.3.1 Two-Dimensional Analytical No-Load Magnetic Field Computations 39

2.4 Results ......................................................... 47

2.4.1 Slotless Armature: Optimal Thickness of Windings .......... 47

2.4.2 Halbach vs. Conventional Rotor: Torque Comparison .......... 49

2.5 Intuitive Explanation of Torque Production with Halbach and Conventional Magnet Arrays ................................................. 51
2.6 A Halbach Winding for Stator Design? ........................................ 53
2.7 Conclusions ........................................................................... 54

3 Prototype Wheel Motor: Design and Construction .................. 56
  3.1 Introduction ........................................................................ 56
  3.2 Desired Mechanical and Performance Specifications ............ 59
  3.3 Integration of Motor into Wheel ........................................... 60
  3.4 Electromagnetic Modeling .................................................. 62
    3.4.1 No-Load Airgap Flux Density ...................................... 63
    3.4.2 Armature Reaction Fields ............................................. 68
    3.4.3 Torque ...................................................................... 74
    3.4.4 Back-EMF .................................................................. 74
    3.4.5 Back-Iron Flux Density ................................................. 74
    3.4.6 Eddy Current and Hysteresis Losses .............................. 76
    3.4.7 Motor Efficiency .......................................................... 77
  3.5 Results ................................................................................ 79
    3.5.1 Number of Pole-Pairs .................................................. 79
    3.5.2 Optimal Winding Thickness .......................................... 79
    3.5.3 Performance Parameters of Prototype motor ................. 80
    3.5.4 Thermal Performance ................................................... 85

4 Wide Speed Range Operation of PMSMs Without Flux Weakening 91
  4.1 Introduction ........................................................................ 91
  4.2 Standard 3-Phase Voltage Source Inverter ............................ 95
  4.3 Novel 3-Phase Voltage Source Inverter ................................. 99
  4.4 Energy Storage Requirements ............................................. 101
  4.5 Conversion Efficiency ......................................................... 103
  4.6 Installed Volt-Ampere Capacity .......................................... 104
  4.7 Snubbing Requirements ...................................................... 104
  4.8 Component Cost ............................................................... 104
  4.9 Control Complexity ........................................................... 105
# 5 Prototype Wheel Motor: Power Electronics and Control

## 5.1 Introduction

## 5.2 Power Circuit

- 5.2.1 Semiconductor Component Specification
- 5.2.2 Energy Storage Component Specification
- 5.2.3 Semiconductor Losses
- 5.2.4 Inverter Efficiency

## 5.3 Control

- 5.3.1 Average Models
- 5.3.2 Digital Controllers

## 5.4 Experimental Circuit Implementation

- 5.4.1 Commutation circuits
- 5.4.2 Gate Drive Circuits
- 5.4.3 Analog Instrumentation Circuits
- 5.4.4 Microcontroller and Peripheral Digital Circuits

# 6 Prototype Fabrication and Experimental Results

## 6.1 Inverter Construction

## 6.2 Motor Construction

## 6.3 No-Load Back-EMF

## 6.4 Control Loop Simulations

## 6.5 Characterization of Motor-Inverter Efficiency

- 6.5.1 Thermal Performance

## 6.6 Summary

# 7 Summary and Conclusions

## 7.1 Thesis Summary

## 7.2 Thesis Conclusions

## 7.3 Recommendations for Future Work
A  Circuit Schematics  

B  GAL22v10 Programming Files  
   B.1  Phase Current Selection Logic  
   B.2  IGBT Switching Logic  
      B.2.1  Switching Logic for Current-Mode Control of Boost Converter  
      B.2.2  Switching Logic for Current-Mode Control of Boost Converter  
   B.3  R-S Latches for User Interface  
   B.4  Miscellaneous Logic  
      B.4.1  Current-Mode Control of Boost Converter  
      B.4.2  Duty-Cycle Control of Boost Converter  

C  Determination of Feedback Gains  
   C.1  Buck Converter Feedback Gains  
   C.2  Boost Converter Feedback Gains  

D  Maple Source Code  
   D.1  Halbach vs. Conventional Torque Comparison  
   D.2  Slotted Motor Design  
   D.3  Inverter Design  

E  Microcode
# List of Figures

1-1  CARB mandated emission standards. .............................................. 18  
1-2  Losses in an EV powertrain during motoring. ............................... 23  
1-3  Four-wheel-drive all-electric and hybrid powertrain with wheel motors. 24  
1-4  Demagnetization characteristics of typical PM materials. ................ 26  
1-5  Effect of temperature on the demagnetization characteristics of Nd-Fe-B magnets. .......................................................... 27  
2-1  Magnetization pattern in continuous and discrete Halbach arrays. ........ 32  
2-2  Motor operating points under Urban and Highway Drive Schedules. .... 34  
2-3  Intuitive illustration of flux distribution in conventional and Halbach magnet arrays. ......................................................... 36  
2-4  Finite element simulation of flux distributions of conventional and Halbach magnet arrangements. .............................................. 38  
2-5  Analytical model of generalized magnetic circuit. ........................... 39  
2-6  Radial and tangential flux distribution within armature windings for conventional and Halbach arrangements. ......................... 46  
2-7  Optimal thickness of armature windings for high torque production. .... 48  
2-8  Torque production in Halbach and conventional rotor magnet arrays with and without a permeable rotor backing ................................. 50  
2-9  Intuitive explanation of torque production in with non-magnetic rotor backing. .............................................................................. 52  
2-10 Intuitive explanation of torque production in with permeable rotor backing. ............................................................................ 54  
3-1  Photograph of the Synergy 2010 hybrid electric vehicle. (Courtesy: Ford Motor Company.) .......................................................... 57
4-5 RMS current through bus capacitors over the entire region of operation of the motor. .......................... 97
4-6 Resistor-Capacitor-Diode snubber for high-current IGBTs. .................. 98
4-7 New inverter topology. ........................................ 99
4-8 Regions of operation of buck and boost sections of inverter. .............. 100
4-9 RMS current through bus capacitors in the buck mode of operation. .... 102
4-10 Capacitive snubber for low-current IGBTs. .............................. 105

5-1 Inverter topology. ............................................... 108
5-2 Operating boundaries of buck and boost converters ....................... 111
5-3 Variation of inverter's input current and input voltage with motor current and back-emf. .......................... 112
5-4 Typical switching waveforms and associated energy loss for a hard-switched half-bridge. .............................. 114
5-5 Inverter's semiconductor switching and conduction losses. ............... 115
5-6 Simulated inverter efficiency. .................................... 116
5-7 Average circuit models of buck and boost converters driving wheel motor. 118
5-8 General block diagram of buck and boost control loops. ................. 120
5-9 Desired region for pole placement ................................ 122
5-10 Schematic diagram of inverter implementation. .......................... 124
5-11 Hall sensor - microcontroller interface circuitry ......................... 125
5-12 Truth table for motor commutation and phase selection. ............... 126
5-13 Gate-drive interface circuitry. .................................... 127
5-14 Bus voltage and Boost output voltage sensing circuit. .................. 129
5-15 Boost inductor current sensing circuit. ............................... 130
5-16 Phase current sensing and selection circuitry. ........................... 131
5-17 Connectivity and data flow between the MC68HC16Z1 and its peripherals. 132
5-18 DAC post-processing circuit and current-mode PWM implementation. 133
5-19 Schematic of user-input keypad. .................................. 134
5-20 Simplified flow chart of microprocessor control code. .................. 135
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-1</td>
<td>Schematic diagram of experimental setup.</td>
<td>136</td>
</tr>
<tr>
<td>6-2</td>
<td>Photographs of experimental setup.</td>
<td>137</td>
</tr>
<tr>
<td>6-3</td>
<td>Prototype inverter at various stages of construction.</td>
<td>139</td>
</tr>
<tr>
<td>6-4</td>
<td>Prototype motor at various stages of construction.</td>
<td>141</td>
</tr>
<tr>
<td>6-5</td>
<td>Prototype motor at various stages of construction.</td>
<td>142</td>
</tr>
<tr>
<td>6-6</td>
<td>Variation of no-load peak phase-to-neutral voltage with motor speed.</td>
<td>145</td>
</tr>
<tr>
<td>6-7</td>
<td>No-load phase-to-neutral back-emf waveforms at three different speeds.</td>
<td>146</td>
</tr>
<tr>
<td>6-8</td>
<td>Simulink block diagram and simulated step response of the buck control loop.</td>
<td>147</td>
</tr>
<tr>
<td>6-9</td>
<td>Simulink block diagram and simulated step response of the boost control loop.</td>
<td>148</td>
</tr>
<tr>
<td>6-10</td>
<td>Sample phase-to-neutral voltage and corresponding phase current for buck control loop.</td>
<td>150</td>
</tr>
<tr>
<td>6-11</td>
<td>No-load torques needed to spin the dynamometer-motor assembly at various RPMs.</td>
<td>152</td>
</tr>
<tr>
<td>6-12</td>
<td>Sample inverter input voltage and current waveforms.</td>
<td>153</td>
</tr>
<tr>
<td>6-13</td>
<td>Measured efficiency of the inverter motor combination in percent.</td>
<td>160</td>
</tr>
<tr>
<td>6-14</td>
<td>Computed efficiency of motor in percent.</td>
<td>161</td>
</tr>
<tr>
<td>6-15</td>
<td>Computed efficiency of inverter in percent.</td>
<td>162</td>
</tr>
<tr>
<td>6-16</td>
<td>Inverter input bus voltage in Volts.</td>
<td>163</td>
</tr>
<tr>
<td>6-17</td>
<td>Inverter input power in kW.</td>
<td>164</td>
</tr>
<tr>
<td>6-18</td>
<td>Average armature winding temperature in deg. C.</td>
<td>165</td>
</tr>
<tr>
<td>6-19</td>
<td>Ohmic power dissipation in armature windings in Watts.</td>
<td>166</td>
</tr>
<tr>
<td>6-20</td>
<td>Estimated core loss in stator steel in Watts.</td>
<td>167</td>
</tr>
<tr>
<td>6-21</td>
<td>Total heat generated in motor in Watts.</td>
<td>168</td>
</tr>
<tr>
<td>A-1</td>
<td>Circuitry on gate-drive board.</td>
<td>175</td>
</tr>
<tr>
<td>A-2</td>
<td>Voltage sensing circuitry.</td>
<td>176</td>
</tr>
<tr>
<td>A-3</td>
<td>Boost inductor current processing circuitry.</td>
<td>177</td>
</tr>
<tr>
<td>A-4</td>
<td>Motor phase current processing circuitry.</td>
<td>178</td>
</tr>
<tr>
<td>A-5</td>
<td>Gate-Array Logic chips and power supplies for analog/digital board.</td>
<td>179</td>
</tr>
</tbody>
</table>

Chapter 1

Introduction

1.1 Background

Electric vehicles (EVs) have existed since the late 1800s, and were a dominant means of transportation in the early 1900s [1, 2]. In the late 1920s to early 1930s battery-powered EVs, with their limited energy storage capability and hence limited range, were displaced by the internal-combustion-engine vehicles (ICEVs), and therefore EVs ceased to be a competitive means of transportation.

The increased cost of petroleum during the oil crisis in the early 1970s and concern for national energy security brought attention to electric and hybrid vehicles as a potentially cost-competitive alternative means of transportation. In the United States for example, the Electric and Hybrid Vehicle Research, Development and Demonstration Act of 1976 was passed by the U.S. Department of Energy (DOE), and was amended in 1978 to provide funding for research and development projects for EVs. A few all-electric test vehicles including the first and second generation Electric TransaXel (ETX-I/II), and the ETV-I/II, were developed under the DOE electric and hybrid vehicles program [2]. Outside the United States, many prototype EVs were developed. Some efforts in Japan, Hong Kong, and Europe are documented in [3], [4], and [5] respectively.

To date EVs have not been able to displace ICE vehicles for a number of reasons including the higher energy density and lower cost of petroleum compared to that of electrochemical batteries, and the ease and flexibility of re-fuelling ICE vehicles. Current EV
batteries are expensive and slow to charge, and an extensive battery charging infrastructure comparable to gasoline filling stations for ICE vehicles has yet to be developed. Furthermore, the high price of current EVs coupled with their limited range, lower acceleration and lower attainable top-speed makes the EV an unattractive choice in comparison to their rival ICE vehicles. Even for short trips such as intra-city driving and short-distance commuting where range is not an issue, these expensive EVs have to compete against older ICE vehicles that are usually acquired at lower cost. Thus EVs remain a tough sell to the public at large.

However, in October 1990, the California Air Resources Board (CARB) introduced legislation aimed at reducing urban atmospheric pollution. This legislation, which mandates that 2% of all the cars and light trucks sold by 1998 in the state of California be zero-emission vehicles (ZEVs) at the tailpipe rekindled interest in the development of battery-powered EVs. The CARB sales quota for ZEVs is scheduled to rise to 10% by 2003 (see Figure 1-1). One year after the CARB legislation was passed, nine other states in the northeast, including Massachusetts, New York and the District of Columbia voted to adopt the CARB regulations. It is estimated that the mandated sales for ZEVs in the United States will be about 300,000 by the turn of the century [6]. ZEVs have therefore moved from the research phase to the development of production prototypes by Ford, General Motors, and Chrysler, as well as other small EV companies, such as U.S. Electricar in California and Solectria Corporation in Massachusetts, who retrofit ICE vehicles with electric motors. Examples of the current state-of-the-art EVs include the Ford Ecostar, GM Impact, Chrysler G-Van and TEVan in the United States, and the BMW-E1/E2, Nissan FEV and Tepco Iza in Europe and Japan, respectively.

Unfortunately, as the year 1998 rapidly approaches, U.S. Automakers have yet to find an affordable traction battery that can give ZEVs a sufficient range to make them an attractive choice for the general consumer. Recognizing this problem, CARB in July 1995 proposed to amend its mandated ZEV quota to include a new class of vehicles called equivalent zero-emission vehicles (EZEVs). These EZEVs are allowed to have up to only 10% of the emission of ultralow-emission vehicles (ULEVs) which are to be introduced in 1997. The rational is that, EZEVs would produce no more nitrogen oxides (NOx) and non-methane organic gases (NMOG) than would the electric power plants in the California South Coast
Air Basin in charging battery-powered ZEVs. The provision for some tailpipe emissions allow for hybrid vehicles, which permit the vehicle to be propelled by a combination of an efficient heat engine and an electric motor, to qualify as EZEVs. These hybrid vehicles would also earn partial ZEV credit depending on their all-electric range and other safeguards designed into the vehicle to preserve its EZEV status such as prevention of the use of the heat engine to charge the battery or power accessories in the all-electric mode [7].

Adding to the momentum to develop EVs as a competitive means of transportation in the United States is the Partnership for a New Generation of Vehicles (PNGV) initiative, launched by President Clinton and Vice President Gore with the Chief Executive Officers
of the Big Three U.S. Automakers on September 29, 1993. The PNGV initiative is aimed at developing technologies that would strengthen U.S. competitiveness in the world automotive market. To this end, the U.S. government and the U.S. Council of Automotive Research (USCAR) have launched development efforts to develop a mid-size family sedan that would achieve up to three times the fuel efficiency of today’s comparable vehicle such as the 1994 Chrysler Concorde, Ford Taurus and Chevrolet Lumina [8]. This new vehicle, dubbed the Supercar, aims at achieving 80 miles per gallon of gasoline and meeting ULEV emission levels of 0.125 gal/mile of hydrocarbons (HC), 1.7 g/mile of carbon monoxide (CO) and 0.2 g/mile of nitrogen oxides (NOx) at 100,000 miles while complying with other Clean Air Act requirements. These requirements make the all-electric or hybrid vehicle a strong candidate for the future car. In evaluating the fuel efficiency of the EV, one gallon of gasoline is considered equivalent to 114,132 BTUs (33.4 kWh) which, after processing through a typical power plant that is 40% efficient results in 13 kWh of available energy. Thus, the EV must achieve the formidable energy efficiency of at least 0.16 kWh/mile in order to meet the PNGV fuel efficiency target. Note that this energy efficiency is for the powertrain alone and does not include the energy required to run the so-called “hotel loads” in the vehicle, such as the air-conditioning, passenger compartment heaters, window defrosters etc. Current prototype EVs achieve a powertrain efficiency of about 0.32 kWh/mile.

1.2 Technological Challenges

Many technological innovations still remain to be achieved in order to improve the performance of EVs. First on the list is obviously the development of high-capacity low-weight traction batteries or other forms of on-board energy storage for extended driving range. The next is the development of a more energy-efficient, high-performance powertrain.

1.2.1 Energy Storage

One of the major challenges facing EV development is the lack of a high-energy and high-power density on-board energy storage medium. This lack of adequate energy storage severely limits the range of all-electric vehicles, powered by electrochemical batteries today,
to about 100 miles per charge [9]. For EVs to gain wide public acceptance, significant progress must be made in this domain.

Assuming a total mileage of about 150,000 miles over a vehicle lifetime, and a range of about 150 miles per charge, the traction battery must survive about 1000 charge-discharge cycles. According to the U.S. Advanced Battery Consortium, EV batteries will need to achieve a specific energy of about 200 Wh/kg at C/3\(^1\) discharge rate, a specific power of 400 W/kg (80% depth of discharge per 30 seconds), a recharge time of 3 to 6 hours, a battery lifetime of about 10 years and cost less than $100 per kWh [9].

Lead-acid has been the battery technology of choice in many EV prototype vehicles such as the GM Impact because it is a low-cost, commercially available and a proven battery technology. However, these batteries suffer from low specific energy and specific power of about 35 Wh/kg and 120 W/kg respectively, low life-cycle of approximately 750 cycles, and high recharge times of about 8-12 hours [10]. Sodium-Sulphur (NaS) battery technology, pioneered by Ford Motor Company, has also found widespread use in prototype EVs, including the Ford Ecostar and BMW-E1/E2, and is considered to be a very promising battery technology due to its high specific energy and specific power. NaS batteries today achieve about 81 Wh/kg and 152 W/kg, with a theoretical projection for energy storage targeted at 150 Wh/kg. Their recharge time of 6-8 hours is significantly lower than that of Lead-acid, however, this high operating temperatures of about 350 - 380°C pose safety problems. Additionally, they have a low life cycle of only 592 cycles [13]. Nickel-based batteries, such as Nickel-Cadmium, Nickel-Iron, Nickel-Zinc, Nickel-Metal-Hydride also have high energy densities and high peak powers but are very costly. Among the Nickel-based batteries, Nickel-Metal-Hydride has shown the most potential. Ovonic Battery Company has achieved a specific energy density of 95 Wh/kg with Nickel-Metal-Hydride technology and there are talks of a joint venture between Toyota Motor Corporation and Matsushita Electric Industrial Company to develop, produce and market Nickel-Metal-Hydride batteries. Thus Nickel-Metal-Hydride batteries may be the EV battery of choice in the mid-term. Although Nickel-Cadmium is a proven battery technology, its toxic nature disqualifies it as being a contender for a viable EV battery. Lithium-based batteries are

\(^1\)C/3 discharge rate: the current that would discharge a fully charged battery in 3 hours
also considered to have great potential due to their high energy densities and short recharge times. In fact, Lithium-Aluminum/Iron-Disulfide batteries boast of a recharge time of “minutes” but are currently only experimental. In the long term, Lithium-polymer batteries with a specific energy density of four to five times that of contemporary lead-acid batteries are expected to be produced for $100 per kWh, according to the U.S. Advanced Battery Consortium.

Although it is not easy to predict which battery technology will eventually emerge as a winner, the metrics of a winner are well established, namely high energy and power densities, short recharge times, long cycle life, environmental friendliness and, of course, low cost. Other forms of energy storage including fuel cells, ultracapacitors, and electromechanical flywheels are also being researched [30]. Containment for on-board storage of hydrogen is currently a big issue for fuel cells. Commercially available ultracapacitors today have a low specific energy density of about 1-2 Wh/kg and need significant improvement in this domain. Prototype flywheel systems with installed specific energy and power densities of 5 Wh/kg and 375 W/kg have been demonstrated in transit buses, however, significant work remains to be done to increase specific energy density, provide lightweight containment, simplify overall system integration and reduce cost [8].

1.2.2 Current EV Powertrains

EV powertrains can be broadly classified into two categories: all-electric and hybrid powertrains. All-electric vehicles derive all of their propulsion power from an electric motor whereas a hybrid vehicle can accept power from a combination of two sources, typically an electric motor and an internal combustion engine.

The hybrid electric vehicle (HEV) concept is also not a new one. In fact, U.S. patent records show that hybrids have existed since the early 1900s [12]. With hybrids, the availability of an alternative source of propulsion energy, other than an electrochemical battery, helps to increase the range of the vehicle and control emissions at the same time since the auxiliary ICE can be operated at its most efficient point. Thus, if the CARB-proposed EZEV category that accommodates hybrid vehicles is implemented, a near term
goal of an EV that can offer competitive performance to an ICE vehicle while meeting strict emission levels may be achieved. A disadvantage of hybrid powertrains, though, is the added cost and complexity deriving from the addition of the second prime mover.

The EV industry has not as yet adopted a particular type of motor drive as a standard and a great deal of experimentation still continues. Electric powertrains can employ a single (or multiple) high-speed central motor(s) with mechanical reduction gears or direct-drive wheel motors. There are also a host of motor technologies to choose from such as induction, switched-reluctance, wound-field and permanent-magnet. The GM-Impact, Nissan FEV and Ford Ecostar, for example, use three-phase induction motors with reduction gears. The G-Van and TExVan use series wound dc motors while the BMW-E1, ETX-II and Tepco IZA use permanent-magnet motors [9, 11]. Different forms of mechanical integration and packaging abound such as the integration of reduction gears and planetary differentials into a single transaxle as in the Ford Ecostar.

1.3 Towards a more Efficient Electric Powertrain

Permanent-magnet (PM) motors, despite their potentially catastrophic failure modes, are gaining popularity in traction drives due to the low weight and high efficiencies that these motors offer. The use of a single high-speed central motor with mechanical transmission offer advantages in weight and cost reduction of the motors and simplicity in vehicle control, since only one motor needs to be controlled. However, one suffers about 10-15\% losses in mechanical transmissions (Figure 1-2), and the mechanical linkages from the central motor-drive to wheels increase the part count and the complexity of the powertrain.

Figure 1-2 show the losses from battery to wheel in a typical electric vehicle during motoring. Significant simplicity in the mechanical design and packaging of the electric vehicle can be obtained by integrating the drive motor into the wheel as shown in Figure 1-3, and thereby eliminating the need for shafts, transaxles, differentials, clutches and reduction gears. The resulting powertrain can be more efficient, as the losses from the mechanical transmission are completely eliminated, and the vehicle is easier to assemble and manufacture due to the reduction in mechanical part count. In the EV driven by
wheel motors, the rated torque is equally distributed among the drive wheels. Each of the traction motors has its own independent drive and each can be controlled electronically to yield the torque-speed characteristics of a motor-gearbox combination. A differential is implemented electronically by commanding the same torque at different wheels rotating at different speeds. Full starting torque can be applied directly at the wheel and the motors can be controlled to yield a smoother drive at low speeds. Furthermore, auxiliary hydraulic actuators for traction control such as antilock braking systems can be eliminated since these functions can be implemented electronically.

Of all the prototype EVs cited in Section 1.1, only the Tepco IZA is advertised to have used a wheel motor arrangement, but little data on the powertrain performance and cost savings derived from such a wheel motor arrangement is available in the open literature.

A wheel motor based traction system, however, poses the unique challenge of designing a high-torque, low-mass motor, in the restricted mechanical envelope offered by the hub of a wheel, that would operate efficiently over the wide speed range of the vehicle. The low motor mass is needed to reduce the unsprung weight of the vehicle, as this weight affects vehicle suspension design and hence ride quality. The motor must operate in a dirty and harsh environment characterized by a wide range of ambient temperatures, and large shock and vibration forces. Induction motors are ill-suited for this application due to the difficulty
Figure 1-3: Four-wheel-drive all-electric and hybrid powertrain with wheel motors.

of removing heat from the rotor, which is a rotating member. Switched-reluctance motors require a very small airgap to produce high torques and hence raise a survivability issue with regard to their ability to withstand the vibration forces at the wheel; their high level of acoustic noise notwithstanding. Permanent-magnet synchronous motors (PMSMs) do not suffer from the above disadvantages since there are no significant rotor losses and large airgaps are possible. In view of the foregoing concerns about other motor technologies, our studies in this thesis were restricted to PMSMs as they do not suffer from the above deficiencies.
1.4 Enabling Technologies

Improvements in the performance of conventional EV powertrains that use induction motors, over the last decade, have mainly been in the area of drive and control electronics. The availability of high-power, fast-switching, and low-loss semiconductor devices, coupled with super-fast microprocessors and floating point digital signal processors (DSPs) have allowed for better waveform synthesis and control for induction motor drives. The story for PM powertrains is a bit more extensive. Here, in addition to the above improvements in drive and control electronics, the availability of new high-energy permanent magnet materials have also stimulated a number of novel PMSM designs [14, 15]. Despite these significant improvements in technical performance, device and material costs still remain high and a major breakthrough in cost reduction is still needed to drive down the cost of EV powertrains.

1.4.1 Permanent Magnets

Figure 1-4 shows the demagnetization curves for current commercially available PM materials. Until the early 1970s, the only permanent magnets that were available to PM motor designers were Alnico and Ceramic Ferrites. Then came along rare-earth magnet materials such as Samarium Cobalt (Sm-Co) in the 70s followed by Neodymium-Iron-Boron (Nd-Fe-B) in the early 80s.

Nd-Fe-B magnets lead in terms of energy product, remanent flux density and coercive force. Commercially available Nd-Fe-B magnets have energy products that range from 27-42 MGOe (215 - 334 kJ/m³) and improvements are still being made in this area towards the projected theoretical maximum energy product of 64 MGOe (509 kJ/m³) [19]. These magnets are finding increasing use in old applications such as voice coil motors for computer disk drives, high-performance small-sized synchronous and stepper motors, as well as new applications that include Magnetic Resonance Imaging (MRI) systems, undulators or electron wigglers for the production of high quality synchrotron radiation, eddy-current retarder brakes for large motor vehicles such as trucks and busses, as well as EV propulsion motors. The main disadvantages of Nd-Fe-B magnets are their relatively high temperature
dependence of remanent flux and coercive force as shown in Figure 1-5, and their poor resistance to corrosion. The corrosion problem is often solved by employing one of a host of proven coating technologies including, ion vapor deposition of aluminum and/or chromate coating of 7-19 μm thickness, electrodeposition of an epoxy coating of 20-30 μm and electroplating the magnets with about 10-20 μm of nickel. Magnet manufacturers are also producing new grades of Nd-Dy-Fe-Co-B-Mo sintered magnets with improved temperature stability [16, 19]. Sm-Co, the predecessor to Nd-Fe-B, does not suffer from the corrosion and temperature problems described above, but is about 2-3 times as expensive as Nd-Fe-B. Current bulk prices for Nd-Fe-B on the U.S. market are about $80 - $100 USD per lb, and there are reports that magnets with similar performance (but less professional finishing) to that obtainable on the U.S. market, can be obtained from the Tsinghua University in China [18].

1.4.2 Power Semiconductor Switches

The Insulated Gate Bipolar Transistor (IGBT) is gradually becoming the power semiconductor switch of choice in EV drives. The IGBT combines the desirable high voltage and current switching capabilities of the bipolar transistor as well as the high input impedance
Figure 1-5: Effect of temperature on the demagnetization characteristics of Nd-Fe-B magnets.

and fast switching speed of the Metal-Oxide-Semiconductor Field-Effect Transistor. Its square safe operating area reduces its snubbing requirements in hard-switched inverters. With switching times of less than a microsecond, third generation IGBTs are capable of PWM operation up to 20 kHz, thus lowering ripple current filtration requirements and acoustic noise. Design and development of high-power variable-speed drives have been further simplified with the introduction of Intelligent Power Modules (Intellimods\textsuperscript{TM}) by Powerex, in alliance with Mitsubishi Electric [20]. These Intellimods\textsuperscript{TM} integrate gate drives and other protection circuitry with third-generation IGBTs and soft recovery anti-parallel diodes in one module, simplifying the interface of the power switches to inverter control circuitry. Intellimods\textsuperscript{TM} are currently available as singles, duals (two IGBTs connected in a half-bridge configuration), or six-packs (six IGBTs connected as three half bridges on a common bus). The rated current of these devices range from 10 A to 600 A with a rated collector-to-emitter voltage of 600 V or 1200 V, thus making them appropriate for the design of multi-kiloWatt inverters. Intellimods\textsuperscript{TM} with rated collector-to-emitter voltage of 1500 V are about to be introduced.
1.5 Thesis Objectives and Contributions

This thesis seeks to accomplish three major objectives. The first objective is to develop, for the first time, a prototype 15 kW (20 hp) high-performance direct-drive PM wheel motor, novel IGBT-based power-electronic inverter and a microprocessor controller for the propulsion of a battery-powered or hybrid mid-size family sedan. The motor employs Nd-Fe-B magnets, is capable of producing a constant torque of 450 Nm up to 325 rpm, and 15 kW of constant power up to a top speed of 1300 rpm. The active mass of the motor is only 25 kg, compared to the competition which has a mass of 53 kg [21]. The motor-inverter combination has been tested on a dynamometer in a laboratory environment and the drive efficiencies from battery to wheel are estimated at 85 % and 92 % over the Federal Urban Drive Schedule (FUDDS) and the Federal Highway Drive Schedule (FHDS) respectively.

To arrive at an economical motor design that uses a minimum amount of surface-mounted magnets, a detailed study of the torque production characteristics of a new class of surface-mounted PM motors called Halbach motors, which to date have not been widely used in the electric machines community, was performed. The motivation for this study was driven by the search for novel electromechanical arrangements with the high torque densities (i.e. low mass) and high efficiencies required by an EV wheel motor. Through this endeavor, we gain a quantitative and intuitive understanding of the operation of this class of PM machines and outlined their merits vis a vis conventional surface-mounted motors. We have been able to answer the question as to when and why one should consider using a Halbach magnet arrangement in the design of PMSMs. We learn that a slotted stator with square-wave windings and a conventional surface-mounted rotor magnet arrangement, where magnetization vector rotates 180° from one rotor pole to the next, yields a more economic wheel motor than a slotless stator with Halbach magnets.

The third objective is to develop a novel power-electronic inverter that is particularly suited for PMSMs with a high torque constant, and hence high back-emf constant, that require wide-speed-range operation. This novel inverter topology consists of standard three-phase voltage source inverter, modified to include a bidirectional boost converter in its dc link. This topology allows PMSMs to be designed for wide speed range operation.
without the need to introduce rotor saliency in the motor. Efficient extended speed range operation without field-weakening or phase advance, which carry the penalties of reduced efficiency and increased torque ripple, is made possible. For an IGBT inverter operation from a fixed dc bus, the conduction losses are reduced if the output power is delivered at a high voltage and hence low current since the on-state voltages of the IGBT and its antiparallel diode are fairly constant with current. Furthermore, high voltage, low-current inverters have fewer electrolytic dc bus filter capacitors, smaller current-related voltage overshoots, less conduction losses and hence smaller heatsinks, smaller conductors and high efficiency. Since the semiconductor switches are rated to withstand the highest motor back-emf, catastrophic failure modes that will result in large currents injected by the motor into the traction battery are avoided.

1.6 Thesis Organization

Recognizing that the ability to reduce the no-load losses in a traction motor is key towards motor efficiency maximization over both Urban and City drive cycles, Chapter 2 sets out to explore and compare the torque production characteristics of slotless Halbach slotless conventional surface-mounted PMSMs and their suitability for application in a wheel motor powertrain, since slotless motors have significantly lower no-load losses compared to their slotted counterparts. Unfortunately we learn that, to produce the high torques necessary for a direct-drive wheel motor, slotless machines require a substantial amount of high-energy Nd-Fe-B magnets, thus making the resulting motors uneconomical. Chapter 3 therefore deals with the design of a slotted motor which can produce the high torques necessary with reduced magnet mass. The magnet arrangement is of the conventional type since with reduced magnet mass and a permeable rotor backing, the conventional magnet arrangement is preferred.

In seeking to increase the efficiency of the power-electronic inverter, achieve a wide-speed-range for motor operation without inefficient flux-weakening, and also prevent the catastrophic failure modes associated with inverter reset, Chapter 4 examines the benefits of driving the motor with a novel high-voltage/low-current inverter, compared to a low-
voltage/high-current inverter. In Chapter 5, we design the inverter and the microprocessor controller for the wheel motor designed in Chapter 3.

The prototype wheel motor and inverter are built and tested in Chapter 6. The motor has a back-emf constant of 7.26 V/rad-s. An efficiency map under the low-speed region of the torque-speed envelope is provided.

Finally, Chapter 7 presents a brief summary of the ground covered in this thesis and conclusions drawn from this experience, and it lays out the roadmap for future work.
Chapter 2

Torque Production in Slotless Halbach and Conventional Permanent-Magnet Motors: A Comparative Analysis

2.1 Introduction

The elimination of intermediate gearing in direct-drive applications requires the drive motor to produce the total load torque electromagnetically. High torques in an electric motor can be achieved by increasing the magnetic loading, electric loading and force lever arm. Volume constraints and rotor inertia often limit the maximum lever arm of the motor and the electric loading is limited by the maximum power dissipation in the armature windings as well as the coercive force of the rotor magnet material, since one has to guard against demagnetization of the rotor magnets. The magnetic loading is also limited by saturation of the rotor and stator back iron and losses in the stator back iron. If expensive magnet materials such as Nd-Fe-B or Sm-Co are to be used for high-performance, low-weight machines, then magnet cost and hence the volume or mass of magnet material becomes a constraint for economical designs.

The choice of rotor magnet arrangement is an important initial step in the design of permanent-magnet synchronous motors. Numerous rotor magnet configurations, such as
buried (interior) tangential, radial and inclined magnets exist in addition to their commonly used surface-mounted counterparts, and their relative merits have been compared in [22, 23] using finite-element simulations. The Halbach permanent-magnet arrangement, named after its inventor Klaus Halbach, has traditionally been used to produce harmonically pure magnetic fields for undulators or electron wigglers for particle accelerators, free electron laser systems, and synchrotron radiators [24] - [28]. Recently the Halbach array is finding use in electric machines including linear motors for wafer transport, high-speed energy storage flywheels and direct-drive wheel motors for electric vehicle propulsion [29, 30, 31].

The ideal Halbach magnet arrangement is one whose magnetization rotates continuously in a sinusoidal fashion as schematically shown in Figure 2-1(a). Since a continuously rotating magnetization is difficult to achieve in practice, the ideal array is often approximated with discrete sets of magnets. In [32], Marinescu et al. compared the airgap flux of a discrete Halbach array where the magnetization vector rotates 60° (3 magnet segments per pole) and 45° (4 magnet segments per pole) between magnet pieces to that of a conventional radially-magnetized surface-mounted arrangement. The results in [32] showed that the multipolar Halbach array is capable of producing higher peak airgap flux and hence machine torque.

To date, the general question of when and why one should use a Halbach rotor in the design of electric machines has not been addressed. In this chapter we study and compare
the torque production capability of Halbach and conventional rotor magnet arrangements on a slotless armature. Furthermore, to simplify the fabrication of the rotor magnet array it is useful to consider a Halbach array which consists of only radial and azimuthal magnetizations, where the magnetization vector rotates 90° between magnet pieces as shown in Figure 2-1(b). Such a rotor arrangement is easier to manufacture since in applications with high number of pole-pairs and large rotor diameter, such as a wheel motor, the radial and azimuthal magnets can be cut out of the same magnetized block.

Figure 2-2 shows a gray-scale representation of the proportion of time the drive motor spends at the operating points underneath its normalized torque-speed curve under the FUDS and the FHDS [38]. Darker shades in the Figure represent more frequent operation. Our motivation for studying a slotless armature stems from noting from Figure 2-2 that the ability of the EV propulsion motor to produce low torques efficiently is key in efficiency maximization over both the city and highway drive cycles, as shown by the shaded areas in Figure 2-2. Slotless motors are desirable in this respect since the stator teeth, which often carry high flux densities that contribute significantly to the no-load motor losses, are eliminated. The armature conductors in a slotless machine however, are not shielded by the stator teeth and hence can exhibit significant eddy-current losses. The eddy-current loss, fortunately, is proportional to the fourth power of the diameter of the armature conductors and hence can be significantly reduced by reducing the strand size of the conductors, with the lowest strand size set by manufacturing constraints[33]. The conductors must be properly transposed so that each strand experiences the same average magnetic field. Thus the armature windings are often made out of Litz wire. With a slotless armature, rotor pole-face losses associated with stator slot harmonics are also eliminated, as well as torque ripple associated with magnet cogging which can be troublesome at low vehicle speeds. EV drives notwithstanding, slotless motors are becoming serious candidates for high speed applications such as machine tools and generators, and also in other applications where low cogging torque is necessary, such as direct drive servo positioning, and have been the subject of many recent papers [34] - [37]. In studying the Halbach and conventional arrangement for the rotor magnets, our goal is to determine which of the two magnet arrangements produces the highest torque per mass of magnet material used, subject to the same armature
Figure 2-2: Motor operating points under Urban and Highway Drive Schedules.

- Produce Low Torques Efficiently
- Efficient Operation Crucial at Medium to High Speeds
- Reduce No-Load Losses in Motor
- Consider slotless motor
current excitation. We also seek to determine regions of operation where each magnet arrangement performs best, in terms of peak torque per magnet mass, thus aiding the motor designer to choose appropriately between the two magnet arrays. We study the two cases where the rotor backing is made of a highly-permeable material such as magnetic steel or a non-magnetic material such as synthetic fibers for mechanical reasons [30]. The volume of magnets as well as the power dissipation in the armature windings are kept the same for each study.

We find that in machines with a permeable rotor backing, the conventional arrangement produces a higher torque up to a certain magnet thickness after which the Halbach arrangement excels. For machines without a permeable rotor backing, the Halbach arrangement always produces higher torque. We develop simple intuitive explanations, in terms of magnetic charges, that explain this result, and enable one to predict the torque production capability of these machines without resorting to somewhat cumbersome mathematics.

2.2 The Halbach Array: An Intuitive Understanding of Spatial Flux Distribution

The Halbach array has traditionally been used to produce harmonically pure magnetic fields for use in undulators and electron wigglers. It has been shown in [29] that for the ideal Halbach array, the magnetic field intensity vanishes on one side of the array, and for a square block array such as the one shown in Figure 2-3(d), the spatial fundamental field is canceled on one side of the array while the field intensity on the other side enhanced by $\sqrt{2}$.

In this section we present an intuitive model that explains the “one-sidedness” of discrete Halbach arrays. For this purpose, it is useful to think of magnet blocks as being composed of paired magnetic charges. Magnetic fields emanate from positive magnetic charges and terminate on negative charges. Furthermore, field lines from positive charges terminate on the closest “free” negative charge. With these simple rules, the field lines from the conventional magnet array are constructed, as shown in Figure 2-3(a).

The field lines from the Halbach array are constructed via superposition. We first
Figure 2-3: Intuitive illustration of flux distribution in conventional and Halbach magnet arrays.
separate the magnet blocks into a radial (vertical for a linear motor) array and an azimuthal (horizontal for a linear motor) array as shown in Figure 2-3(b) and (c). By applying the simple rules outlined above, one can construct the field lines for the individual arrays as shown in Figure 2-3(d). The spatial flux distribution of the combined array is the sum of the flux distributions for the individual arrays. Note that there is field cancellation on one side of the array while the fields on the other side of the array reinforce. Which side of the array the cancellation takes place depends on the direction of rotation of the magnets. Figure 2-4 shows a finite-element simulation of the magnetic flux distribution of a conventional and Halbach magnet arrangement for a 10 pole-pair cylindrical rotor, using the 2-D finite-element analysis program QuickField\textsuperscript{1}. The rotor backing is non-magnetic in both simulations. The flux distribution in Figure 2-4(b) shows the one-sidedness of the Halbach arrangement vis-a-vis that of the conventional arrangement in Figure 2-4(a).

2.3 Torque Computation

In this section, we study the torque production characteristics of conventional and Halbach arrays with and without a permeable rotor backing. The armature used for this comparison is slotless, due to its desirable attributes of low no-load losses and absence of magnet cogging. The winding consists of a three-phase set wound in a square-wave fashion, of which two phases out of three are excited at a time with rectangular currents. We have explored various winding placement patterns and numbers of phases, under the constraint of fixed ohmic power dissipation in the armature and have learned that a 3-phase square-wave excitation with only two phases on at a time yields high torque while keeping the inverter simple and the cost of the semiconductor switches under control. In this study, the outer diameter of the motor is held constant at its maximum value (as dictated by the application) and our objective in optimization is therefore to determine, for a given volume of magnets, which configuration of rotor magnets produces the highest torque per ohmic power dissipated in the armature windings. The machine torque is computed by integrating the product of the

\textsuperscript{1}QuickField is a registered trade mark of Tera Analysis Company, 17114 Bircher Street, Granada Hills, CA 91344
(a) Conventional Magnet Arrangement on Non-Magnetic Backing

(b) Halbach Magnet Arrangement on Non-Magnetic Backing

Figure 2-4: Finite element simulation of flux distributions of conventional and Halbach magnet arrangements.
radial no-load magnetic flux and the armature current density over the volume occupied by the armature conductors. For the torque computation, we assume that the coercive force of the magnets is large enough to avoid demagnetization.

2.3.1 Two-Dimensional Analytical No-Load Magnetic Field Computations

Figure 2-5 shows a cross-section of the magnetic circuit of the motor to be analyzed, where we have ignored end fringing effects by making the structure infinitely long into the figure. This generalized magnetic circuit can be used to analyze the no-load magnetic field density distribution for both Halbach and conventional motors, with and without a permeable rotor backing. To analyze designs without a permeable rotor back-iron, the dimensions $R_{ri}$
and \( R_{ro} \) are set to infinity. For designs with permeable rotor backing, \( R_{ri} \) is set to \( R_1 \). The parameter \( \beta \) represents the ratio of the pole arc of the radial magnets to the pole pitch. Thus by varying \( \beta \) from zero to one, the full range of Halbach rotor designs with various pole-arc to pole-pitch ratios can be studied. For example, \( \beta = 1 \) corresponds to a conventional design with full-pitch magnets since this value of \( \beta \) leaves no room for the azimuthal magnets. On the other hand, \( \beta = 0 \) corresponds to a rotor magnet arrangement with only azimuthal magnetization. As will be seen later on in this Chapter, we shall compare the torque comparison of Halbach and conventional magnets arrays under the constraint of fixed magnet mass. A parameter \( \delta \) will be introduced, which can be used to turn off the azimuthal magnetization to permit non full-pitch conventional magnet arrays to be studied. For Halbach designs \( \delta = 1 \), and for conventional designs, \( \delta = 0 \). For \( \delta = 1 \), the resulting Halbach magnet arrangement corresponds to that shown in Figure 2-5, and for \( \delta = 0 \), the resulting conventional arrangement corresponds to that shown in Figure 2-5 with the azimuthal magnet segments removed.

In our magnetic circuit analysis, the recoil permeability, \( \mu_r \), of the magnets is assumed to be unity; an assumption which is not bad for rare-earth magnets such as Nd-Fe-B (\( \mu_r = 1.05 \)) or Sm-Co (\( \mu_r = 1.02 \)). We are interested in computing the magnetic flux density in the armature windings in order to compute torque. This region occupied by the armature conductors, together with the running clearance constitute region \( c \) in Figure 2-5. The magnetic field intensity in the magnets (region \( b \)) is needed to assess possible demagnetization of the azimuthal magnets. In our analyses, it will be assumed that the sizes of all permeable back-iron are adequately chosen such that infinite permeability can be assumed and magnetic saturation can be ignored.

A. Representation of magnetic field sources

The radially-oriented magnets are assumed to be radially magnetized with a magnetization vector given by \( M_{rad} = M_0 R_m / r \hat{r} \), and the magnetization vector for the azimuthal magnets is given by \( M_{az} = M_0 R_m / \hat{\phi} \), where \( M_0 \) is the maximum magnetization and is related to
the remanent flux density $B_{r0}$ by $M_0 = B_{r0}/\mu_0^2$. The remanent flux $B_{r0}$ has a negative temperature coefficient and is related to the magnet temperature in °C by $T_{mag}$ by the equation

$$B_{r0}(T_{mag}) = B_{r20} \left[ 1 - \frac{0.11(T_{mag} - 20)}{100} \right]$$

(2.1)

where $B_{r20}$ is the magnet remanent flux at 20°C. The assumption of radial (instead of parallel) magnetization simplifies greatly our analysis, and does not detract from the generality of our results since for machines with high numbers of pole-pairs, the difference between the flux distributions for radial and parallel magnets is practically negligible [39].

The magnetic flux distributions in the three regions ((a), (b), and (c)) are determined via superposition. We first solve for the fields from the radial magnets and add them to those from the azimuthal magnets. The magnetization vectors for the radial and azimuthal magnets can be written in terms of their Fourier components as

$$M_{rad} = \sum_{k \text{ odd}}^{\infty} \frac{R_m}{r} \frac{4M_0}{k\pi} \cdot (-1)^{\frac{k-1}{2}} \sin k\beta \pi \cdot \sin kN\phi \hat{r}$$

(2.2)

$$M_{az} = \sum_{k \text{ odd}}^{\infty} \frac{R_m}{r} \frac{4M_0\delta}{k\pi} \cdot (-1)^{\frac{k-1}{2}} \cos k\beta \pi \cdot \cos kN\phi \hat{\phi}$$

(2.3)

where $k$ is the harmonic number and $N$ is the number of pole pairs. The parameter $\delta$ is set to unity for the Halbach arrangement and zero to turn off the azimuthal magnetization for the conventional arrangement. Notice that the divergence of each of the Fourier components of $M_{rad}$ is zero, therefore the magnetic field in all three regions (a), (b), and (c) are governed by Laplace’s Equation, for the radially-oriented magnets. For the azimuthal magnets, the Fourier components of the magnetization vector are not divergence-free even though the total magnetization vector $M_{az}$ has no divergence. Therefore the solution for the magnetic field intensity in the magnets (region (b)) contains a particular solution from Poisson’s Equation, in addition to a Laplacian homogeneous solution which is used to satisfy the boundary conditions.

---

2 Vector quantities are represented with bold typeface throughout this thesis.
B. Boundary conditions

We solve for the magnetic fields via use of a scalar potential $\psi$ which obeys Laplace's Equation $\nabla^2 \psi = 0$ in cylindrical coordinates. The magnetic field intensity is given by $\mathbf{H} = -\nabla \psi$. The permeabilities of the stator and rotor steel, when present, are assumed to be infinite for ease of illustration in this thesis. At the boundaries $r = R_{ri}$ and $r = R_{so}$, the corresponding scalar potentials $\psi_{(a)}$ and $\psi_{(c)}$ vanish due to the infinite permeability of the steels. At the boundaries $r = R_1$ and $r = R_m$, we require the tangential magnetic field $H^\phi$ and the radial component of the magnetic flux density $B^r$ to be continuous, noting that within the magnets, the magnetic flux density $\mathbf{B} = \mu_0 (\mathbf{H} + \mathbf{M})$. It is also useful to note that continuity of $H^\phi$ at a boundary is equivalent to the scalar potential $\psi$ being continuous at that boundary.

C. Solutions

The general solutions for the magnetic scalar potentials in the three regions are of the form

\[
\psi_{(a), \text{rad}} = \sum_{k \text{ odd}}^{\infty} C_{1rad} \left[ \left( \frac{r}{R_{ri}} \right)^{kN} - \left( \frac{R_{ri}}{r} \right)^{kN} \right] \cdot \sin kN \phi \tag{2.4}
\]

\[
\psi_{(b), \text{rad}} = \sum_{k \text{ odd}}^{\infty} \left[ C_{2rad} r^{kN} - C_{3rad} r^{-kN} \right] \cdot \sin kN \phi \tag{2.5}
\]

\[
\psi_{(c), \text{rad}} = \sum_{k \text{ odd}}^{\infty} C_{4rad} \left[ \left( \frac{r}{R_{so}} \right)^{kN} - \left( \frac{R_{so}}{r} \right)^{kN} \right] \cdot \sin kN \phi \tag{2.6}
\]

for the radially-oriented magnets and

\[
\psi_{(a), \text{az}} = \sum_{k \text{ odd}}^{\infty} C_{1az} \left[ \left( \frac{r}{R_{ri}} \right)^{kN} - \left( \frac{R_{ri}}{r} \right)^{kN} \right] \cdot \sin kN \phi \tag{2.7}
\]

\[
\psi_{(b), \text{az}} = \sum_{k \text{ odd}}^{\infty} \frac{M_{az} R_m}{kN} + C_{2az} r^{kN} - C_{3az} r^{-kN} \cdot \sin kN \phi \tag{2.8}
\]

\[
\psi_{(c), \text{az}} = \sum_{k \text{ odd}}^{\infty} C_{4az} \left[ \left( \frac{r}{R_{so}} \right)^{kN} - \left( \frac{R_{so}}{r} \right)^{kN} \right] \cdot \sin kN \phi \tag{2.9}
\]

for the azimuthally-oriented magnets. Note that the solutions for $\psi_{(a)}$ and $\psi_{(c)}$ are chosen such that they vanish at $r = R_{ri}$ and $r = R_{so}$ respectively, and that the first term in Equation 2.8 is the particular solution. The equations that must be solved in view of the the boundary
conditions outlined in the previous section are:

\[
\begin{align*}
\psi(a)_{rad} \bigg|_{r = R_1} &= \psi(b)_{rad} \bigg|_{r = R_1} \\
\psi(a)_{az} \bigg|_{r = R_1} &= \psi(b)_{az} \bigg|_{r = R_1} \\
\psi(b)_{rad} \bigg|_{r = R_m} &= \psi(c)_{rad} \bigg|_{r = R_m} \\
\psi(b)_{az} \bigg|_{r = R_m} &= \psi(c)_{az} \bigg|_{r = R_m}
\end{align*}
\]

\[
\begin{align*}
B^r(a)_{rad} \bigg|_{r = R_1} &= B^r(b)_{rad} \bigg|_{r = R_1} \\
B^r(a)_{az} \bigg|_{r = R_1} &= B^r(b)_{az} \bigg|_{r = R_1} \\
B^r(b)_{rad} \bigg|_{r = R_m} &= B^r(c)_{rad} \bigg|_{r = R_m} \\
B^r(b)_{az} \bigg|_{r = R_m} &= B^r(c)_{az} \bigg|_{r = R_m}
\end{align*}
\]

Solving Equations (2.10) - (2.17) for the arbitrary constants \( C_{1rad} - C_{4rad} \), and \( C_{1az} - C_{4az} \), via the symbolic manipulation package Maple V3 yield

\[
C_{1rad} = \frac{M_{rad} R_m R_{ti}^{kN}}{2kN \left(R_{ti}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(R_{m}^{kN} - R_{so}^{2kN} R_{ti}^{-kN} - R_{m}^{kN} + R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[
C_{2rad} = \frac{M_{rad} R_m}{2kN \left(R_{ti}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(R_{m}^{kN} - R_{m}^{kN} + R_{so}^{2kN} R_{m}^{-kN} - R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[
C_{3rad} = \frac{M_{rad} R_m}{2kN \left(R_{ti}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN} - R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN} - R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN} + R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[
C_{4rad} = \frac{M_{rad} R_m R_{so}^{kN}}{2kN \left(R_{so}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(R_{m}^{kN} - R_{m}^{kN} + R_{so}^{2kN} R_{m}^{-kN} - R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[
C_{1az} = \frac{M_{az} R_m R_{so}^{kN}}{2kN \left(R_{so}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(-R_{so}^{2kN} R_{so}^{-kN} + R_{m}^{kN} - R_{m}^{kN} + R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[
C_{2az} = \frac{M_{az} R_m}{2kN \left(R_{so}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(R_{m}^{kN} - R_{m}^{kN} - R_{so}^{2kN} R_{m}^{-kN} + R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[
C_{3az} = \frac{M_{az} R_m}{2kN \left(R_{so}^{2kN} - R_{so}^{2kN}\right)} \cdot \left(R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN} - R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN} - R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN} + R_{so}^{2kN} R_{so}^{2kN} R_{m}^{-kN}\right)
\]

\[\text{Source code for computation of unknown constants is attached in Appendix D.1. Maple V is a registered trademark of Waterloo Maple Inc, 450 Phillip St., Waterloo, ON, Canada, N2L5J5.}\]
\[ C_{4az} = \frac{M_{az} R_m R_{s0}^{kN}}{2kN \left( R_{r1}^{2kN} - R_{r2}^{2kN} \right)} \cdot \left( P_{r1}^{2kN} R_m^{-kN} + R_m^{kN} - R_1^{kN} - R_{r1}^{2kN} R_1^{-kN} \right) \]  

\[ (2.25) \]

where

\[ M_{az} = \frac{4M_0 \delta}{k\pi} \cdot (-1)^{\frac{k-1}{2}} \cos k\beta \pi \]  

\[ (2.26) \]

and

\[ M_{rad} = \frac{4M_0}{k\pi} \cdot (-1)^{\frac{k-1}{2}} \sin k\beta \pi \]  

\[ (2.27) \]

The total magnetic scalar potential in each region is the sum of the contributions from the radial and azimuthal magnetizations and are therefore given by

\[ \psi_{(a),tot} = \psi_{(a),rad} + \psi_{(a),az} \]  

\[ (2.28) \]

\[ \psi_{(b),tot} = \psi_{(b),rad} + \psi_{(b),az} \]  

\[ (2.29) \]

\[ \psi_{(c),tot} = \psi_{(c),rad} + \psi_{(c),az} \]  

\[ (2.30) \]

The airgap magnetic flux density \( B_{ag} \), which is used to compute machine torque, is given by

\[ B_{ag} = -\mu_0 \nabla \psi_{(c),tot} \]  

\[ (2.31) \]

D. Torque production under fixed magnet volume

For a fair comparison of the torque production capability of Halbach and conventional magnet arrays, we need to fix the volume (and hence the mass and cost) of the magnets used in each design. Thus, given an outer radius \( R_1 \) for the magnet array, the inner radius \( R_m \) can be calculated as

\[ R_m = \sqrt{R_1^2 - \frac{M_{mag}}{\frac{\pi \rho_{mag} l_{mag}}{\beta + \delta(1 - \beta)}}} \]  

\[ (2.32) \]

where \( \rho_{mag} \) is the magnet density, \( M_{mag} \) is the magnet mass, and \( l_{mag} \) is the magnet length. Remember that \( \delta = 1 \) for Halbach designs and \( \delta = 0 \) for conventional designs. For a given armature current excitation, the peak motor torque within a commutation interval \( T \) is computed by integrating the product of the radial component of the no-load airgap magnetic
flux density $B_{ag}^r$ and the armature current density over the volume occupied by the armature conductors as

$$T = -2NW_{eff} \int_{R_{so}}^{R_{Cu}} \int_{\frac{z}{2N}}^{\frac{zN}{2}} J_0 \cdot B_{ag}^r r^2 d\phi dr \hat{\phi}$$  \hspace{1cm} (2.33)

where $J_0$ is the armature current density, $B_{ag}^r$ is the radial component of the no-load magnetic flux density as defined in Equation 2.31, $W_{eff}$ is the effective length of the motor, which is the stack length reduced by a fringing factor, and $R_{so}$ and $R_{Cu}$ refer to the inner and outer radii of the annulus formed by the armature windings, as shown in Figure 2-5. The expression in Equation 2.33 is computed using Maple V. Note that this torque expression is a function of the magnet mass, the magnet pole-arc to pole-pitch ratio $\beta$, the thickness of copper ($R_{Cu} - R_{so}$), and the Halbach/conventional parameter $\delta$.

To illustrate the torque production characteristics of slotless Halbach and conventional motors, we consider the design of a slotless motor for a possible direct-drive wheel motor application. The outer diameter of the magnets, $R_1$, is fixed to 19.6 cm and the active length of the motor $W_{eff}$ is 6 cm, such that the motor can fit in the hub of a 19-inch wheel. The ohmic ($I^2R$) power dissipated in the armature windings is fixed to 2.5 kW and a copper packing factor of 0.7 is assumed. The motor has 10 pole-pairs and a running clearance of 0.625 mm (25 mils) exists between the magnets and the armature windings. The remanent flux of the magnets at 20°C is 1.2 T, and the magnet temperature is 110°C, reducing the remanent flux to 1.08 T.

Figure 2-6 shows the radial and tangential components of the the airgap flux density within the armature windings for both the conventional and Halbach arrangements. For these plots, the magnet mass is 16 kg, the pole-arc to pole-pitch ratio $\beta$ is 0.75, and winding thickness ($R_{Cu} - R_{so}$) is set to 2 cm to highlight the decay of the airgap flux. Both rotor and stator backings are magnetic with infinite permeability, and the extent of the plots is one pole-pair in the azimuthal direction. First, note that the tangential flux is zero at outer periphery of the stator ($R_{so}$) as expected from the infinitely permeable stator. Secondly, the fields decay away from the magnets with the higher harmonics decaying faster, as expected from Laplace’s Equation. These observations serve as a check of the correctness of the analytic solutions. The interesting observations to be made from these figures are that
Figure 2.6: Radial and tangential flux distribution within armature windings for conventional and Halbach arrangements.
the field strength for the Halbach array is higher than that of the conventional array, and furthermore, that some high peaks exist in the radial field distribution of the Halbach array at the areas where the azimuthal magnets abut the radial magnets. The existence of these high peaks may suggest that one should put the armature windings in the area underneath the peaks. However these peaks decay rapidly and in both the Halbach and conventional arrangements higher torques are produced when the windings are placed underneath the center of the pole; thus producing torque from the peak of the fundamental component of the rotor’s magnetic field.

2.4 Results

2.4.1 Slotless Armature: Optimal Thickness of Windings

One interesting characteristic of slotless motors is the existence of an optimal thickness of armature windings to produce maximum torque per ohmic power dissipated in the armature windings. The existence of this optimal winding thickness can be understood by considering the variation of the airgap magnetic flux density and the current density in the armature windings as the winding thickness is varied. Figure 2-7(a) shows the variation of the current density in the armature windings with the thickness of the windings under the constraint of constant ohmic power dissipation in the armature. For fixed $I^2R$ dissipation, the allowable current density decreases with the winding thickness. Plotted in Figure 2-7(b) is the torque per current density in the windings, which is the integral of the radial component of the no-load airgap magnetic flux density over the volume occupied by the armature conductors. With zero winding thickness, the torque per current density is zero since the integration is over zero volume. The product of the two plots which represents the peak motor torque within a commutation interval is shown in Figure 2-7(c). This figure indicates that for illustrative design under consideration, the optimal thickness of armature windings is 0.8 cm. Above this thickness of copper, the available torque degrades gradually and the excess copper adds unnecessarily to the weight and cost of the motor. Furthermore, if the armature windings are to be cooled by circulating coolant fluid at the inner surface to the stator
Figure 2-7: Optimal thickness of armature windings for high torque production.
back-iron, the excess copper increases the thermal resistance of the stator windings. In general, the optimal thickness of copper would decrease with the number of pole-pairs, $N$, mainly because the magnetic field decays faster within the airgap for high $N$.

### 2.4.2 Halbach vs. Conventional Rotor: Torque Comparison

Figure 2-8 shows 3-D plots of the peak motor torque within a commutation interval against the magnet mass $M_{mag}$ and the magnet pole-arc to pole-pitch ratio $\beta$. In both plots, the motor parameters are the same as in Section 2.4.1 and the armature winding thickness is set to its optimal value of 0.8 cm for the given number of pole-pairs. The plot in Figure 2-8(a) compares conventional and Halbach rotor designs with a permeable rotor backing while that in Figure 2-8(b) compares the same designs with a non-permeable rotor backing. Note that for fixed $R_1$, as the magnet volume increases, $R_{Cu}$ and $R_{so}$ decrease to accommodate the additional magnets while maintaining the optimal copper thickness. The usefulness of these plots is to determine, for a given magnet mass and hence magnet cost, which rotor magnet configuration yields the highest torque and what is the corresponding optimal $\beta$.

The plot of Figure 2-8(a) shows that for designs with a permeable rotor backing, the conventional array produces higher torque than the Halbach array when the magnets are thin. In this region of the design space, the magnets are not full pitch (i.e. $\beta < 1$) and higher torque is produced by shaving off magnets from the pole edges where the magnets abut, and using this magnet material to increase the radial length of the pole magnets. For thick magnets, however, the Halbach array produces higher torque irrespective of $\beta$. Note that the optimum $\beta$ changes with the thickness (mass and hence volume and cost) of the magnets and $\beta = 1$ corresponds to a conventional arrangement where the magnets are full-pitch. For $\beta = 1$, the Halbach becomes equivalent to the conventional design with full-pitch magnets as expected. The plot in Figure 2-8(b), however, shows that for designs with a non-permeable rotor backing, the Halbach array always produces higher torque than the conventional array. The use of a permeable rotor backing in both the Halbach and conventional design allows the same torque to be produced with less magnets as the corresponding design with a non-permeable rotor backing. The above conclusions hold
(a) Achievable peak torques with magnetic rotor backing.

(b) Achievable peak torques with non-magnetic rotor backing.

Figure 2-8: Torque production in Halbach and conventional rotor magnet arrays with and without a permeable rotor backing.
in general and are not specific to the illustrative example in Figure 2-8. In general, when the magnets become very thick, the designs with and without a permeable rotor backing approach each other but the torque “saturation” level of the Halbach rotor is higher than the conventional rotor.

The plots in Figure 2-8 are for the peak torque within a commutation interval but the average torque also exhibits similar dependence the magnet mass and $\beta$, except that the optimum $\beta$ and the crossover magnet thickness (for magnetically backed rotors) are slightly different from those of the peak torque. Notice that in both designs (with and without a permeable rotor backing) the highest torque within the fixed mechanical envelope is achieved with the Halbach design. However this high torque is only achievable with a large volume of magnets which may be prohibitive from an economic point of view. Thus from an economic point of view, a slotless motor is not a good choice for a wheel motor which may be required to produce about 450 Nm of torque, as magnet mass in excess of 16 kg would be required. By shortening the electromagnetic airgap via armature slotting, once can reduce the magnet mass necessary to establish the required airgap flux density and forgo the benefits of high efficiency and low cogging associated with slotless motors.

2.5 Intuitive Explanation of Torque Production with Halbach and Conventional Magnet Arrays

To gain an intuitive understanding of the foregoing results, it is useful to think of the source of electromagnetic force not as the cross product of the radial component of the no-load airgap flux density and the armature current density but rather as the tangential component of the magnetic field from the armature conductors acting on the magnetic charges. Figure 2-9 shows the magnetic field lines from two excited phases of the armature windings. Also shown in the figure are the magnetic charges associated with the magnet arrays; the conventional array is shown in Figure 2-9(a) and the Halbach array is shown in Figure 2-9(b). In both figures, the volume of magnets is kept the same and the rotor magnets and armature conductors are aligned to produce maximum torque.
Figure 2-9: Intuitive explanation of torque production in with non-magnetic rotor backing.

The “good” charges interact with the tangential component of armature conductor magnetic field to produce positive torque. The “bad” charges, on the other hand, interact with the tangential magnetic field to produce retarding torque. Note that the strength of the magnetic field interacting with the “bad” charges is low for reasonable magnet thicknesses so there is a net positive torque. The shaded charges in Figure 2-9(a) are the “useless” charges. The magnetic field lines interacting with these charges are effectively radial so they produce no tangential force. By cutting the magnet pieces containing these “useless” charges and rotating them 90° as shown in Figure 2-9(b), these charges now experience a substantial tangential magnetic field, and hence become “good” charges producing positive
torque. This is the benefit of the Halbach arrangement. Once again, note that the magnet volume remains unchanged.

A similar explanation can be offered for designs with a permeable rotor backing by considering the reflection of the magnetic charges in the steel via the method of images. Note that with both rotor and stator back-iron present, there are infinite reflections. However the overall effect of magnetic backing can be understood by focusing on the primary reflection of the magnetic charges through the rotor back-iron. Figure 2-10(a) shows that a permeable magnetic backing effectively doubles the length of the magnets with radial magnetization. The effect is to further remove the negative charges which produce retarding torque from regions where the armature reaction is strong. Thus there is a significant boost in motor torque when thin radial magnets are backed with steel. The azimuthal magnets, however, reflect the wrong way as shown in Figure 2-10(b). The image charges tend to cancel the otherwise “good” charges. Thus for thin magnets, the “good” charges created by introducing azimuthal magnetization may be completely cancelled by their image charges and hence produce no additional torque. In this situation, one does better by using the azimuthal magnet material to lengthen the radial magnets. When the radial magnets become thick enough such that the effect of magnetic backing is negligible, then one does better by introducing azimuthal magnetization since with thick azimuthal magnets, there is only partial cancellation of charges at the magnet-steel interface, and some “good” charges now appear in regions where there is strong tangential armature reaction. The above intuitive explanations reinforce the generality of our conclusions about the torque production characteristics of Halbach and conventional magnet arrays.

2.6 A Halbach Winding for Stator Design?

The Halbach array without a permeable backing has two desirable attributes: (i) the flux density pattern is single-sided, which makes it attractive in applications where shielding of magnetic fields is important, and (ii) the magnitude of the flux density on the reinforced side is factor of $\sqrt{2}$ higher than that of a conventional array for the same volume of magnets, making it attractive when magnet cost is an issue. One may therefore be tempted to believe
(a) Reflection of radial magnetic charges via method of images

(b) Reflection of azimuthal magnetic charges via method of images

Figure 2-10: Intuitive explanation of torque production in with permeable rotor backing.

that in iron-less motors, a stator winding configured in a Halbach fashion may produce a higher armature reaction than a conventional winding. Unfortunately, to obtain the single-sidedness of magnetic flux density offered by a Halbach magnet array with windings, power is dissipated in windings whose sole function is to cancel the flux on one side of the array. Thus the flux density on the reinforced side, per unit power dissipated in the windings, is lower than that of a conventional array. A detailed analysis of a Halbach winding can be found in [40]. Although a Halbach stator winding in not attractive for the design of power-efficient motors, superconducting electromagnets configured in a Halbach fashion may be useful in shielding passenger compartments in a maglev application. [40]

2.7 Conclusions

We conclude that, for permanent-magnet synchronous motors where the application precludes the use of a magnetic back-iron, the Halbach array always produces higher torque than the conventional array, for the same volume of magnets. The use of a magnetic back-iron in both designs increases the achievable torque. However, for magnetically-backed rotors, the conventional array (with an optimized pole-arc to pole-pitch ratio) produces
higher torque than the Halbach array in the regime of low magnet volume and hence thin magnets. When the cost of magnets is not prohibitive, and thick magnets can be used, the Halbach array will produce higher torque within the fixed mechanical envelope.

The optimal $\beta$, in general, depends on the number of excited phases and changes with the magnet volume as was shown in Section 2.4. As $\beta$ approaches unity for Halbach designs, the resulting aspect ratio of the azimuthally-oriented magnets makes them prone to demagnetization even at no-load. This problem gets worse with an increasing number of pole-pairs and magnet volume, as the azimuthal magnets become flat. For magnets with relatively low coercive force such as ceramic Ferrites, demagnetization of the azimuthal magnets severely limits the range of useful $\beta$ and number of pole-pairs. Thus, in optimizing $\beta$ for high torque, one has to guard against demagnetization of the azimuthal magnets even at no-load.

For the direct-drive wheel motor application, the cost of magnets alone preclude the use of a slotless motor despite its attractive features of high efficiency and low cogging. A slotted motor will be a better choice for this application since the reduction of the electromagnetic airgap by the introduction of stator slots reduces the required magnet volume. With thin magnets mounted on a permeable back-iron, a conventional magnet arrangement is preferred to a Halbach array.
Chapter 3

Prototype Wheel Motor: Design and Construction

3.1 Introduction

Figure 3-1 shows a photograph of the Ford Motor Company PNGV concept vehicle, dubbed the Synergy 2010, a futuristic fuel-efficient aerodynamic vehicle with enough room for a family of six. The vehicle is a hybrid electric vehicle that consists of a small 1.0-liter direct-injection, compression-ignited engine that serves as the prime mover for a generator which produces electricity to power four wheel motors through four power electronic inverters shown in Figure 3-2. An energy storage flywheel captures the vehicle’s excess engine and regenerative braking energy which is later used to supply the vehicle’s peak power demands. In this chapter, we will be concerned with the design and construction of a 20 hp PM motor that fits in the hub of a 19-inch wheel shown in Figure 3-2.

Despite the high remanent flux of Nd-Fe-B magnets, a substantial amount of magnets is still needed to establish a high enough magnetic flux density in the armature windings of a slotless motor, even with a Halbach magnet arrangement. The plots of maximum achievable torques in Figure 2-8 show that, with a slotless armature, about 16 kg of magnets is needed to achieve 400 Nm of peak torque within a 19-inch rim. This magnet cost is excessive and does preclude the use of a slotless armature in an economical wheel motor design. Therefore, we take a look at a slotted armature for the design of the prototype wheel motor.
Figure 3.2: Cutaway of the Synergy 2010 hybrid electric vehicle. (Courtesy: Ford Motor Company.)
Figure 3-3: Desired torque-speed envelope of wheel motor.

By slotting the stator, the electromagnetic airgap is reduced and hence thinner magnets mounted on a magnetic back-iron can be used. In Chapter 2, we learned that for thin magnets backed by a permeable back-iron, a conventional magnet arrangement is a better choice. Therefore the wheel motor designed in this Chapter will feature the conventional magnet arrangement. The resulting wheel motor design will be fabricated and tested in Chapter 6.

3.2 Desired Mechanical and Performance Specifications

Figure 3-3 shows the desired torque-speed envelope of the prototype wheel motor. In commanding negative torque, the motor is used as a generator to capture the vehicle’s
braking mechanical energy. A constant peak torque of 450 Nm is desired up to 325 rpm. Above this “corner” rpm, a constant mechanical power of 15 kW (20 hp) is required up to a maximum speed of 1300 rpm. This maximum rpm corresponds to a top road speed of about 100 mph for a 19-inch wheel.

In normal vehicle operation, shock and vibration forces of about 20 g are not uncommon, thus the wheel motor must have a running clearance of at least 0.5 mm (20 mils) for survivability [38]. Allowing for tolerances in machining the magnet arcs, rotor back iron and stator laminations, a running clearance of 0.625 mm (25 mils) is be used.

In a typical midsize sedan, the ratio of sprung to unsprung mass is about 6:1 [42]. Therefore for the PNGV vehicle whose curb weight\(^1\) is about 900 kg, the weight of each wheel motor with its brake assembly and connecting structural members should be less than 37.5 kg if the vehicle suspension is not to be re-designed. For our design, a target mass of 25 kg for the active part of the motor will be used, leaving 12.5 kg for the remaining mechanical structures.

3.3 Integration of Motor into Wheel

Before embarking on the electromagnetic design of a wheel motor, it is necessary to determine the allowable mechanical envelope for the motor and how to mechanically integrate it into the wheel. This design was done in conjunction with Mr. Allan Gale, Mr. Brad Bates and mechanical engineers at Ford Motor Company in Dearborn, MI. Figure 3-4 shows how the PM propulsion motor is integrated with the wheel. The motor occupies an annular region bounded by an outer radius of 206 mm (8.11 in), an inner radius of 160 mm (6.30 in) and an axial length of 96 mm (3.66 in) including end turns. The spacing between the outer periphery of the motor and the rim of the tire is necessary to provide a running clearance since the rim undergoes flexure when negotiating a turn. The hollow area bounded by the inner periphery of the stator back-iron houses a mechanical brake which consists of a standard disc-rotor/caliper combination. The wheel rim and the end-bell of

\(^1\)Curb mass is the mass of an operational vehicle fitted with all standard equipment without a driver or payload
the motor are spoked to facilitate convective heat transfer from the inner periphery of the motor. An inside-out stator-rotor configuration facilitates the transfer of the rotary motion from the rotor to the rim of the tire. An additional benefit of an inside-out design is the elimination of a structural retaining hoop for the rotor magnets since the centrifugal force from the wheel rotation presses the magnets against the rotor back-iron.
3.4 Electromagnetic Modeling

The introduction of stator teeth reduces the electromagnetic airgap of the motor compared to a slotless armature and allows thinner magnets to be used. We learned in Chapter 2 that for thin magnets backed by a permeable rotor, the conventional magnet arrangement is preferred to the Halbach arrangement. Thus the rotor magnets will be arranged in the conventional fashion.

Figure 3-5 shows the cross section of a 3-phase slotted motor with one slot per phase per pole. A small running clearance exists between the magnet arcs and the outer periphery of the stator teeth. The slots have parallel edges to accommodate square wire for increased copper packing factor.

With a running clearance of 0.625 mm and an remanent flux of about 1.1 T, a back of the envelope magnetomotive-force-divider calculation yields a radial airgap flux density of about 0.9 T for a magnet thickness of 3 mm, assuming infinitely permeable rotor and stator back-iron. For equal tooth-to-slot width, the flux in the stator teeth is about 1.8 T, which is close to the saturation flux density of most electrical steels.

One of the distinguishing features between a wheel motor and a conventional motor is that a wheel motor is usually designed to maximize efficiency over a drive cycle, whereas
a conventional motor is designed for efficient operation at full rated load. As a result, conventional motors tend to have a much thicker back-iron that a wheel motor, since the back-iron is sized to carry both the magnet and the peak armature reaction flux efficiently. Motors with thick enough back-iron lend themselves to a linear magnetic circuit analysis where infinite permeability for the steel regions can be assumed. In developing analytical models to guide a first-pass design of the wheel motor, it is assumed that the rotor and stator back-iron are thick enough to permit the assumption of infinite permeability. The stator teeth will, in general, be highly saturated at high electrical loading and hence does require a nonlinear magnetic analysis. The motor is then designed to produce more torque than needed and the thickness of the rotor and stator backings are gradually reduced through experimentation with finite element analysis to reduce motor mass. We underscored in Chapter 2 the need to reduce the no-load losses in the motor for drive cycle efficiency maximization; see Figure 2-2. We therefore use 29-Gage laminations of M-19 nonoriented electrical steel for the stator teeth and back-iron, to reduce eddy-current and hysteresis losses.

3.4.1 No-Load Airgap Flux Density

The ability to account for tooth saturation is important in predicting machine torque, since the permeability of the stator teeth significantly affects the equivalent electromagnetic airgap. In a slotted motor, the magnet flux in the airgap is well guided by the teeth as shown in Figure 3-6, thereby making a one-dimensional magnetic circuit analysis adequate. However, we employ a two-dimensional analysis for the no load airgap flux distribution for consistency with our approach in Chapter 2. In order to include the effects of tooth saturation in the computation of the airgap flux we solve the magnetic circuit of Figure 3-7. The rotor and stator back-iron are assumed to be infinitely permeable.

Figure 3-7 consists of three regions; the region with permanent magnetization denoted as region (a), the free space region that constitutes the running clearance denoted as region (b) and the stator teeth region denoted as region (c). Even though region (c) contains slots, it is modeled as a uniform region. The slotting is taken into account when computing the
magnetic field density in the teeth region to iteratively determine the tooth permeability. We do this by multiplying the flux density in the uniform region by a flux concentration factor which accounts for diversion of magnetic flux away from the slots into the teeth. The permeability of region (c) is related, in a nonlinear manner, to its magnetic flux density as shown in Figure 3-8(b). Therefore, to solve for the magnetic flux density in any of the three regions we need to iteratively solve for a self consistent flux density and permeability in region (c).

As was done in Chapter 2, the radial magnetization vector in region (a) can be modeled as

\[
M_{rad} = \sum_{k \text{ odd}}^{\infty} \frac{R_m}{r} \frac{4M_0}{k \pi} \cdot (-1)^{\frac{k-1}{2}} \sin k \cdot \frac{\beta}{\pi} \cdot \sin k \cdot N \cdot \phi
\]  

(3.1)

where \(M_0\) is the maximum magnetization and is related to the remanent flux density \(B_{r(e)}\) by \(M_0 = B_{r(e)} / \mu_0\), where \(N\) is the number of pole-pairs, \(\beta\) is the pole-arc to pole-pitch ratio and \(k\) is the harmonic number. Note that in this conventional arrangement, there is no azimuthal magnetization. The divergence of each of the Fourier components of \(M_{rad}\) is zero, therefore the magnetic field in all three regions (a), (b), and (c) are governed by Laplace's Equation.

A. Boundary conditions

We denote the Laplacian scalar potentials for the magnet, running clearance, and teeth region as \(\Psi_{mag}^{(a)}, \Psi_{mag}^{(b)},\) and \(\Psi_{mag}^{(c)}\) respectively. Due to the infinite permeability assumption
for the rotor and stator back-iron, the scalar potential vanishes at these two boundaries, i.e.

$$\Psi_{(a)}^{mag} = 0 \text{ at } r = R_{ri}$$  \hspace{1cm} (3.2)

and

$$\Psi_{(c)}^{mag} = 0 \text{ at } r = R_{so} \hspace{1cm} (3.3)$$

The scalar potential and normal magnetic flux density are continuous across $r = R_{mi}$ and $r = R_{cu^*}$, where $R_{cu^*}$ is the physical boundary $R_{cu}$ less 5% of the running clearance to account for the lengthening of the electromagnetic airgap from stator slotting. This is equivalent to modifying the length of the airgap by the Carter coefficient.

**B. Solutions**

The solutions for the magnetic scalar potentials in the three regions are of the form

$$\Psi_{(a)}^{mag} = \sum_{k \text{ odd}}^{\infty} C_k \left[ \left( \frac{r}{R_{ri}} \right)^k \sin \frac{kN}{N} \right]$$  \hspace{1cm} (3.4)
(a) B-H Curve for 29-Gage M-19 Steel

(b) B-μ Curve for 29-Gage M-19 Steel

Figure 3-8: B-H and B-μ curves for 29-Gage M-19 nonoriented steel[45].
\[
\Psi_{(b)}^{\text{mag}} = \sum_{k \text{ odd}}^{\infty} \left[ C_2 r^{kN} - C_3 r^{-kN} \right] \cdot \sin kN\phi
\]  
\[
\Psi_{(c)}^{\text{mag}} = \sum_{k \text{ odd}}^{\infty} C_4 \left[ \left( \frac{r}{R_{so}} \right)^{kN} - \left( \frac{R_{so}}{r} \right)^{kN} \right] \cdot \sin kN\phi
\]  

It is easily checked that Equations (3.2) and (3.3) are automatically satisfied. The unknown arbitrary constants \( C_1 - C_4 \) are determined via Maple V, by ensuring scalar potential and tangential magnetic field intensity continuity across \( r = R_m \) and \( r = R_{Cu} \). The Maple V source code is included in Appendix D. The magnetic field strength in the various regions, \( \mathbf{H}_{(\cdot)} \), is the negative gradient of the corresponding scalar potential. That is,

\[
\mathbf{H}_{(\cdot)}^{\text{mag}} = -\nabla \Psi_{(\cdot)}^{\text{mag}}
\]

and the corresponding magnetic flux densities are given by

\[
\mathbf{B}_{(a)}^{\text{mag}} = \mu_0 (\mathbf{H}_{(a)}^{\text{mag}} + \mathbf{M})
\]  
\[
\mathbf{B}_{(b)}^{\text{mag}} = \mu_0 \mathbf{H}_{(b)}^{\text{mag}}
\]  
\[
\mathbf{B}_{(c)}^{\text{mag}} = \mu_t \mathbf{H}_{(c)}^{\text{mag}}
\]

where the permeability \( \mu_t \) in the teeth region, which is related in a nonlinear fashion to the sum of the magnet and armature reaction flux density, is yet to be determined. If we denote ratio of the slot width to the slot pitch at \( r = R_{so} \) as \( \delta \), then the no-load flux density in a tooth \( \mathbf{B}_{\text{tooth}}^{\text{mag}} \) at a radius \( r \) is given by

\[
\mathbf{B}_{\text{tooth}}^{\text{mag}}(r) = B_{(b),r}^{\text{mag}}(r = R_{Cu}) \cdot \frac{R_{cu}}{r - \delta R_{so}} \frac{1}{CF}
\]

where \( B_{(b),r}^{\text{mag}} \) denotes the radial component of the magnet’s flux density in region (b) and \( CF \) is the flux concentration factor in the tooth.

It is easy to see that for finite ohmic power dissipation, \( \delta = 0 \) produces no torque since there is no room for windings, and \( \delta = 1 \) produces little torque since this configuration corresponds to a slotless armature. The optimum \( \delta \) therefore lies between zero and one, and
through experimentation we learned that $\delta \approx 0.5$ i.e. about optimum for torque production.

### 3.4.2 Armature Reaction Fields

In order to compute the motor torque, we need to compute the magnetic field intensity across the electromagnetic airgap due to current excitation in the stator windings, since it is this component of the stator winding magnetic field that interacts with the rotor magnets to produce torque. Figure 3-9 shows the geometrical structure of the stator and the associated magnetic fields. The stator has one slot per pole per phase, and only two out of the three phases are excited at a time as shown in the figure. A square-wave winding is used because the interaction of trapezoidal currents with trapezoidal flux densities produces more torque than sinusoidal currents with sinusoidal flux for the same rms armature current\[43\].

The slotting of the stator does not make the stator magnetic circuit amenable to a distributed two-dimensional analysis. The traditional approach of converting the total stator ampere-turns into an equivalent surface current over the outer periphery of the teeth severely overestimates machine torque, since tooth saturation and flux leakage across slots are ignored by this approach. A one-dimensional magnetic circuit analysis, that approximately models the effects of tooth saturation and leakage across slots and tooth tips is employed since flux paths of the armature reaction fields are easily traced, as shown in the finite element simulation in Figure 3-9(b). In computing the armature reaction fields, the permanent magnetization is “turned off” but keep in mind that the effective permeability of the teeth is dependent on the total flux through them, which includes the magnet flux. Therefore the solution for the armature reaction fields still contains the permeability of the tooth region $\mu_t$ which is iteratively determined from the sum of the magnet and armature reaction fields. The tooth permeability is computed using the total flux density at one-third the distance from the base of the tooth and the resulting permeability is assumed constant along the whole tooth. This approximation results in an underestimation of motor torque since the tooth permeability is much higher towards the top of the tooth due to tapering resulting from the parallel slots.

Before writing the governing equations and solving for the unknown fields indicated in
Figure 3-9: Stator geometry and armature reaction fields.

Figure 3-9, it is instructive to make a few qualitative comments about the torque production characteristics of the slotted machine as it relates to its geometrical parameters. First, the stator slots cannot be made too deep as a substantial fraction of the H-field from the current excitation leaks across the slots through the winding area. These leakage fields increase with decreasing permeability in the teeth. Secondly, for a given number of pole-pairs, there would be an optimal thickness (depth) of copper that maximizes machine torque per unit $I^2R$ power dissipation in the armature windings since the leakage fields depend on the aspect ratio of the slot. A high number of pole pairs reduce the width of the slot and hence
the depth of the winding, if excessive leakage fields are to be avoided. Thus for a given mechanical envelope, the optimal thickness of copper windings is inversely proportional to the number of poles. This phenomenon is equivalent to what we saw earlier for the slotless machines. Thirdly, a wide slot opening reduces the leakage fields across the tooth tips and therefore increases airgap flux and hence the machine torque. However, a wide slot opening makes the airgap flux nonuniform, and exacerbates the torque ripple and eddy current losses in the magnets due to substantial variation of the airgap permeance from slot to tooth. Therefore the slot opening should be made only wide enough to pass the armature conductors through. To eliminate torque ripple from magnet cogging, the stator laminations are skewed by one slot pitch [44].

The unknown armature reaction fields of interest are, (i) the magnetic field intensity at the base of the stator teeth $H_t^{arm}$, (ii) the leakage magnetic field intensity in the slots at the outer periphery of the windings $H_{cu}^{arm}$, (iii) the leakage magnetic field intensity in the slot opening $H_o^{arm}$, and (iv) the magnetic field intensity in the electromagnetic airgap, $H_g^{arm}$, which consists of the running clearance and the magnet region. These field quantities are denoted as $H_t$, $H_{cu}$, $H_o$, and $H_g$ in Figure 3-9 respectively for clarity. To compute magnitude of these armature reaction fields, we perform contour integration along the three Amperian paths indicated in Figure 3-9(a), and utilize Gauss’s Law for magnetic flux continuity around a stator tooth, as also shown in the Figure. A $1/r$ dependence is assumed for the radial variation of the H-field within the teeth and the airgap. The strength of the leakage field across the slots is proportional to the current which it encloses and therefore increase linearly with the radial distance from the bottom of the slots. Proceeding as outlined above, we obtain the following four equations:

\[
2H_t^{arm} R_{so} \ln \left( \frac{R_{cu}}{R_{so}} \right) + 2H_g^{arm} R_{cu} \ln \left( \frac{R_{ri}}{R_{cu}} \right) = 2J_0 A_{cu} \quad (3.12)
\]

\[
2H_t^{arm} R_{so} \ln \left( \frac{R_{cu}}{R_{so}} \right) + 2H_o^{arm} I_o + 2\frac{\mu_0}{\mu_t} H_o^{arm} I_{tp} = 2J_0 A_{cu} \quad (3.13)
\]

\[
2H_t^{arm} R_{so} \ln \left( \frac{R_{so} + t_{cu}}{R_{so}} \right) + 2H_{cu}^{arm} I_{cu} + 2\frac{\mu_0}{\mu_t} H_{cu} I_{tt} = 2J_0 A_{cu} \quad (3.14)
\]

\[
\mu_0 H_{cu}^{arm} \frac{t_{cu}}{2} + \mu_0 H_o^{arm} \xi + \mu_0 H_g^{arm} I_{tp} = \mu_t H_{ts}^{arm} I_{tb} \quad (3.15)
\]
Figure 3-10: Length of end-turns.

where the dimensions $l_{tp}$, $l_{tb}$, $l_{cu}$, and $l_{tt}$ are given by

$$l_{tp} = \frac{2\pi R_{cu}}{6N} - l_o$$

(3.16)

$$l_{tb} = (1 - \delta) \frac{2\pi R_{so}}{6N}$$

(3.17)

$$l_{cu} = \frac{\delta 2\pi R_{so}}{6N}$$

(3.18)

$$l_{tt} = \frac{2\pi (R_{so} + t_{cu})}{6N} - l_{cu}$$

(3.19)

all of which are indicated in Figure 3-9(a). $A_{cu}$ denotes the area occupied by copper and $\xi$ depends on the slot opening $l_o$. Equations (3.12), (3.13) and (3.14) are associated with Amperian contours across the airgap, across the slot opening, and around the outer periphery of the winding respectively. Equation (3.15) simply states that the flux entering the bottom of the tooth from the stator back iron is equal to the sum of the flux that leaks across the slot through the copper area, the leakage flux across the slot opening and the airgap flux. Equations (3.12 - (3.15) are solved simultaneously for the magnetic field strengths in terms of the unknown tooth permeability $\mu_t$.

Assuming the end turns of the armature windings are arranged to conform to the triangular pattern shown in Figure 3-10, then the axial protrusion of the end turns beyond the stator lamination stack $L_{et}$ is estimated as

$$L_{et} = \frac{\pi (R_{so} + R_{cu})}{2N} \cdot \frac{\delta}{\sqrt{1 - \delta^2}}$$

(3.20)
and the equivalent end-turn length for computing power dissipation, \( L_{et,\text{ohm}} \) corresponds to the hypothenuse, whose length is given by

\[
L_{et,\text{ohm}} = \frac{\pi (R_{so} + R_{cu})}{2N} \cdot \frac{1}{\sqrt{1 - \delta^2}}.
\] (3.21)

The maximum current density \( J_0 \) in the windings is related to the ohmic power dissipation in the armature \( P_{ohm} \) via

\[
J_0 = \sqrt{\frac{\sigma P_{ohm}}{2N \cdot 2A_{cu}(W_{ohm} + 2L_{et,\text{ohm}})}}.
\] (3.22)

where \( W_{ohm} \) is the length of the winding in the slot and \( A_{cu} \) is the area within the slot occupied by copper. The winding length, \( W_{ohm} \), the stack length, \( W_s \), and the skew angle, \( \theta_{sk} \), are related by

\[
W_{ohm} = \frac{W_s}{\cos \theta_{sk}}.
\] (3.23)

where \( \theta_{sk} \) is related to the angular slot pitch and other motor geometrical constants by

\[
\theta_{sk} = \arctan \left[ \frac{(R_{so} + R_{cu})}{2} \cdot \frac{\tan \frac{\pi}{3N}}{W_s} \right].
\] (3.24)

The copper area is computed as

\[
A_{cu} = k_{pf} \frac{\pi R_{so}}{6N} t_{cu}
\] (3.25)

where \( k_{pf} \) is the copper fill factor and \( t_{cu} \) is the thickness of the copper or equivalently, the depth of the slot. The copper conductivity \( \sigma \) is related to the winding temperature \( T \) in °C by

\[
\sigma = 5 \times 10^7 \frac{234.5 + 20}{234.5 + T}.
\] (3.26)

As was mentioned earlier, the total magnetic flux density in the teeth is the sum of the contribution from the magnets and the armature reaction. The magnet flux and armature reaction flux oppose each other in the tooth underneath the leading edge of the magnet and aid each other in the tooth underneath the trailing edge of the magnet as shown in
Figure 3-11: Illustration of aiding and opposing magnet and armature fluxes in stator teeth.

Figure 3-11(c). For motor torque and loss calculations in the teeth, we use the reinforced flux density at one-third the distance from the base of the tooth. Thus \( B_{\text{tooth}} \)

\[
B_{\text{tooth}} = \left[ B_{(b),r}^{\text{mag}} \left( r = R_{cu} \right) - \frac{R_{ru}}{r - \delta R_{su}} \right]_{\text{magnet}} + \left[ \mu_r H_{\text{arm}}^{\text{arm}} R_{su} \right]_{r=R_{su} - \frac{L_{cu}}{2}} \quad . \tag{3.27}
\]
and the tooth permeability $\mu_t$ is iteratively solved to be self consistent with $B_{tooth}$. Note that this flux density can be quite high at full load and therefore the resulting permeability for the tooth can be quite low. This low permeability, assumed for all the teeth, yields a pessimistic estimate of the motor torque. However, keep in mind that since infinite permeability has been assumed for the rotor and stator backings, the actual flux densities and motor torque will be lower than that predicted by our models when the rotor and stator back-iron are reduced via finite element analysis to reduce motor mass.

### 3.4.3 Torque

Once the magnetic-field $H$ in the airgap from the armature reaction is known, the peak machine torque which corresponds to the quadrature rotor-stator alignment shown in Figure 3-9 is computed as

$$
\tau_m = 2N\mu_0 M_0 \int_{R_m}^{R_{ri}} \frac{R_{cu} H_{gs}}{r} \cdot r \, dr \, \Phi
$$

(3.28)

$$
= 2N R_{cu} \mu_0 M_0 H_{gs} \left( \frac{R_{ri} - R_m}{\text{magnet thickness}} \right) \Phi
$$

(3.29)

### 3.4.4 Back-EMF

The peak back-emf of the motor is calculated by equating the output mechanical power to the input electrical power in the absence of losses. Thus the motor torque constant $k_t$ equals the motor back-emf constant $k_e$. Therefore,

$$
k_t = k_e = \frac{\tau_m}{I_m}
$$

(3.30)

where $\tau_m$ is the maximum motor torque and $I_m$ is the motor current.

### 3.4.5 Back-Iron Flux Density

The rotor and stator back-iron carry the fluxes created by the magnets and the armature windings, less the leakage flux in the teeth and copper areas. In normal operation these two fluxes are in quadrature, and the peaks do not occur at the same location, as shown in
Figure 3-11(a) and Figure 3-11(b). The total back-iron flux is therefore approximated by the vector sum of the magnet and current fluxes. The magnet flux in the rotor back-iron $\Phi_{rbi}^{mag}$ is computed by integrating the flux density over one-half a pole face at the inner surface of the rotor back-iron. The corresponding flux in the stator back-iron $\Phi_{sbi}^{mag}$ is similarly computed by integrating the airgap flux density at the outer periphery of the teeth. Thus

$$\Phi_{rbi}^{mag} = \int_0^{\frac{\pi}{2N}} B_{(a),r}^{mag} (r = R_{ri}) R_{ri} \, d\phi \quad (3.31)$$

$$\Phi_{sbi}^{mag} = \int_0^{\frac{\pi}{2N}} B_{(b),r}^{mag} (r = R_{cu}) R_{cu} \, d\phi \quad (3.32)$$

Equation (3.32) assumes that there is no flux leakage across the stator slots, which is not so for heavily saturated teeth, thus Equation (3.32) is a conservative estimate of the magnet component of the flux in the stator back-iron. The contribution to the back-iron flux from the currents acting alone is given by

$$\Phi_{rbi}^{arm} = W_{eff} \mu_0 H_t^{arm} l_{tp} \quad (3.33)$$

$$\Phi_{sbi}^{arm} = W_{eff} \mu_i H_t^{arm} l_{tb} \quad (3.34)$$

Therefore, the total flux density in the rotor $B_{rbi}$ and in the stator $B_{sbi}$ are given by

$$B_{rbi} = \frac{\sqrt{|\Phi_{rbi}^{mag}|^2 + |\Phi_{rbi}^{arm}|^2}}{W_{eff} t_{rbi}} \quad (3.35)$$

$$B_{sbi} = \frac{\sqrt{|\Phi_{sbi}^{mag}|^2 + |\Phi_{sbi}^{arm}|^2}}{W_{eff} t_{sbi}} \quad (3.36)$$

where $t_{sbi}$ is the thickness of the stator back-iron and $t_{rbi}$ is the thickness of the rotor back-iron. The rotor and stator back-iron thicknesses that yield a low enough flux density for the assumption of infinite permeability to be valid tend to be quite large and add unnecessarily to the mass of the motor without significant improvement of the drive cycle efficiencies. Our analytical models help us to determine appropriate dimensions for the outer radius of the stator back-iron $R_{so}$ and the inner radius of the rotor back-iron $R_{ri}$. Through finite element simulations, the thickness of the stator back-iron and rotor back-iron are reduced.
until the desired motor torque is reached.

3.4.6 Eddy Current and Hysteresis Losses

Figure 3-12 shows the core loss data for 29-Gage M-19 nonoriented steel for sinusoidal excitation. To obtain an analytical expression for the core loss per pound \( P_c \), the core manufacturer’s data was fitted to the equation

\[
P_c = \frac{k_e f^2 B^2}{P_{\text{eddy}}} + k_h f \left( \frac{B}{1T} \right)^\alpha \quad (3.37)
\]

using a nonlinear least-squares fit algorithm. In Equation (3.37), \( f \) represents the excitation frequency in Hz and \( B \) is the magnetic flux density within the steel in Tesla. The parameter \( k_e \) models the eddy current losses while \( k_h \) and \( \alpha \) model the hysteresis loss. These parameters were computed to be \( k_e = 32.183 \times 10^{-6} \text{ W/lb/Hz}^2/T^2 \), \( k_h = 10.664 \times 10^{-3} \text{ W/lb/Hz} \), and \( \alpha = 1.793 \). The resulting analytical expression is also plotted on Figure 3-12 for comparison. The core loss obtained from Equation (3.37) show reasonable fit with the manufacturer’s loss data, as shown by the closeness of the circles and corresponding dots in Figure 3-12.

The core loss in the stator teeth \( P_{c,\text{teeth}} \) and stator back-iron \( P_{c,\text{bgi}} \) are given by

\[
P_{c,\text{teeth}} = 2.02 M_{\text{teeth}} \left[ k_e f^2 B_{\text{tooth}}^2 + k_h f B_{\text{tooth}}^\alpha \right] \quad (3.38)
\]
\[
P_{c,\text{bgi}} = 2.02 M_{\text{teeth}} \left[ k_e f^2 B_{\text{bgi}}^2 + k_h f B_{\text{bgi}}^\alpha \right] \quad (3.39)
\]

where \( M_{\text{teeth}} \) and \( M_{\text{bgi}} \) are the masses of the stator teeth and back-iron respectively in kilograms. The rotor back-iron, to first order, does not suffer any losses since it moves in synchronism with the stator flux. Asynchronous losses from azimuthal variations in the airgap permeance and eddy-current losses from commutation (both in magnets and back-iron) are deemed to be small.
3.4.7 Motor Efficiency

The efficiency of the motor $\eta_m$ is given by

$$\eta_m = \frac{\tau_m \omega_m}{\tau_m \omega_m + P_{ohm} + P_c}$$  \hspace{1cm} (3.40)

where $P_{ohm}$ is the dc ohmic loss and $P_c$ is the sum of the eddy current and hysteresis losses in the stator teeth and back-iron. The motor efficiency over a given drive-cycle $\eta_m^{\text{cycle}}$ is given by

$$\eta_m^{\text{cycle}} = \frac{\sum_{\text{cycle}} \tau^{(i)} \omega_m^{(i)} \theta^{(i)}}{\sum_{\text{cycle}} \left[ \tau^{(i)} \omega_m^{(i)} + P_{ohm}^{(i)} + P_c^{(i)} \right] \theta^{(i)}},$$  \hspace{1cm} (3.41)
Figure 3-13: Time spent at various operating point under urban and highway drive cycles[38].

where $t^{(i)}$ denotes the time spent in seconds at the operating point $(i)$ underneath the torque-speed envelope. These numbers are shown in Figure 3-13. Since a PM machine produces negative torques as efficiently as positive torques, the negative torques shown in Figure 3-13 are converted to positive for the drive-cycle efficiency calculations. All these calculations are done with Maple V and the source code is included in Appendix D.2.
3.5 Results

3.5.1 Number of Pole-Pairs

One of the results that stem from our analytical models is that a high number of pole-pairs $N$ is desired for the following reasons. A high number of poles reduces the length of the end turns, and hence the power dissipated in this part of the armature windings that produces no useful torque. The total flux that the rotor and stator back-iron have to carry also reduce with increasing $N$. Thus a high number of pole-pairs is desirable from the point of view of motor mass reduction. The number of poles however cannot be made too high, as fringing in the airgap and leakage in the stator slots become substantial. Another undesirable feature associated with an increased number of poles is the increase in the electrical frequency of the motor which increase hysteresis and eddy current losses, and reduces the time available for motor commutation. Furthermore, for a given number of slots per pole per phase, the stator teeth become very thin and their mechanical integrity become compromised, problems with machining notwithstanding. In view of the foregoing, $N = 10$ was found, through experimentation, to yield an adequate balance between the above conflicting design parameters.

3.5.2 Optimal Winding Thickness

As was alluded to earlier, for a given number of pole-pairs and fixed power dissipation in the armature, there is an optimum thickness of windings beyond which adding extra copper to the motor adds unnecessarily to the weight of the machine. Figure 3-14 shows a plot of peak motor torque with winding thickness $t_{cu}$ for 10 pole-pairs and 2.7 kW ohmic power dissipation in the stator windings. The magnet mass used for Figure 3-14 is 1.7 kg, the stack length is 6 cm and the running clearance is 25 mils (0.625 mm). The inner radius of the rotor $R_{ri}$ (which is the same as the outer radius of the magnets, see Figure 3-7) is set to 19.6 cm and the magnet and winding temperatures are set to 110$^\circ$C. A slot opening of 3.1 mm is assumed because we anticipate using AWG #10 square wire. The jitter of the curve with increasing $t_{cu}$ is due to the decrease in the tooth width at its base and the
Figure 3-14: Variation of peak motor torque with depth of armature windings.

associated reduction in permeability. At flux densities beyond 1.8 T the slope of plot of flux density with relative permeability is reduced (see Figure 3-8(b)) and the algorithm to find a self-consistent operating point along the nonlinear curve does not easily converge so we reduce the acceptable convergence tolerance. Figure 3-14 is useful in deciding how deep to make the slots. For our candidate wheel motor, we pick $t_{cu}$ to be approximately 1.9 cm. Increasing the copper thickness beyond this value degrades motor torque while increasing motor mass. Furthermore, the thermal performance of the motor is compromised since it is difficult to remove heat from the copper in the slots. Figure 3-14 shows that for the given magnet mass, armature excitation and optimal winding thickness, a peak torque of 472 Nm is achievable.

3.5.3 Performance Parameters of Prototype motor

Figure 3-15 shows a transverse and longitudinal section of the prototype wheel motor. Figure 3-16 shows the detailed geometry and dimensions of the stator slots and rotor magnets. The depth of the slots is optimized to produce the highest peak torque per power
Figure 3-15: Transverse and longitudinal section of prototype wheel motor (as designed)
Figure 3-16: Detailed cross-section of motor (as designed).
dissipation in the armature windings. The magnet mass of 1.7 kg, which results in a magnet thickness of 3.44 mm, is the minimum magnet mass needed to achieve 472 Nm of peak torque with 2.7 kW of ohmic power dissipation in the armature at 110°C. The ohmic power dissipation limit of 2.7 kW is chosen for thermal considerations as discussed in Section 3.5.4. The running clearance is set to 25 mils to allow for manufacturing tolerances and survivability of the airgap under road shock and vibration forces. A magnet pole-arc to pole-pith ratio of 0.9 is chosen to reduce the alignment torque ripple [46]. The gap between adjacent magnets is also necessary to facilitate the assembly of the pre-magnetized magnet pieces on the rotor. The width of the slot opening is determined by the wire size which in turn depends on the number of turns per phase. It is set to 3.1 mm to accommodate AWG #10 wire. The choice of the number of turns for the motor has significant implications on the power-electronic inverter and a detailed discussion of this is deferred till the next chapter.

The thickness of the rotor and stator back-iron were determined from finite-element simulations using the finite-element analysis program QuickField. The rotor and stator back-iron were initially made 5 cm thick and the motor torque, as computed by QuickField, was 496 Nm which is in excess of the target motor torque of 450 Nm. This result shows that our analytical modeling underestimates motor torque as expected but it is within 5% of the finite element result. The thickness of the rotor and stator back-iron were then thinned out to 10 cm and the torque dropped to 460 Nm due to saturation of the steel. At this torque value, the finite element iteration was halted, as the motor torque is likely to reduce through imperfections in the fabrication process.

Figure 3-17 lists the key performance parameters as designed including an efficiency map of the candidate wheel motor. The motor has a mass of 25 kg and produces a minimum of 450 Nm of peak torque while dissipating 2.7 kW in the armature windings at 110°C. The motor torque, self and mutual inductances are computed via finite element analysis using the QuickField. The torque computation is performed via evaluating the Maxwell stress tensor on a closed contour in the airgap. The torque and back-emf constants in Figure 3-17(a) are computed at the maximum armature excitation at a temperature of 110°C. Thus these numbers are expected to be higher at light loads and low temperatures.
Motor's Performance Parameters [as designed]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Peak Torque at 110 deg. C</td>
<td>450 Nm</td>
</tr>
<tr>
<td>Max Phase Current at 110 deg. C</td>
<td>75 A</td>
</tr>
<tr>
<td>Minimum Torque Constant at 110 deg. C</td>
<td>6 Nm/A</td>
</tr>
<tr>
<td>Minimum Back-EMF Constant at 110 deg. C</td>
<td>6 V/rad-s</td>
</tr>
<tr>
<td>FHDS Efficiency</td>
<td>91%</td>
</tr>
<tr>
<td>FUDS Efficiency</td>
<td>85%</td>
</tr>
<tr>
<td>Temperature rise at full load</td>
<td>60 deg. C</td>
</tr>
<tr>
<td>Phase-Neutral Inductance</td>
<td>1.5 mH</td>
</tr>
<tr>
<td>Mutual Inductance</td>
<td>0.7 mH</td>
</tr>
<tr>
<td>Leakage Inductance</td>
<td>1.7 mH</td>
</tr>
<tr>
<td>Motor Resistance at 20 deg. C (phase-phase)</td>
<td>0.34 Ohms</td>
</tr>
<tr>
<td>Active Motor Mass</td>
<td>25.1 kg</td>
</tr>
</tbody>
</table>

Rotor: Magnets (1.7 kg), Back-Iron (5.9 kg)
Stator: Teeth (6.6 kg), Back-Iron (4.8 kg), Copper (6.1 kg)

(a) Simulated motor parameters [as designed]

(b) Simulated efficiency map of motor [as designed]

Figure 3-17: Summary of motor's performance parameters and efficiency map.
The self $L_s$, mutual $L_m$ and leakage $L_{lk}$ inductances per turn are computed from two experiments. First one of the three phases is excited with a test current $I_{test}$ the total stored magnetic energy $W_{ms}$ measured. Then two of the phases are excited with the same test current $I_{test}$ and the corresponding stored magnetic energy $W_{mm}$ measured. The measured energies in the above two experiments are related to the test current and the inductances by the equations

\[
W_{ms} = \frac{1}{2} (L_s + L_{lk}) I_{test}^2
\]

\[
W_{mm} = (L_s + L_{lk}) + L_m I_{test}^2.
\]

For a set of three-phase windings displaced by 120°, the mutual inductance is one-half the self-inductance, assuming perfect coupling. Thus we can write relate $L_s$ to $L_m$ by

\[
L_s = 2L_m.
\]

Solving Equations (3.43) - (3.44) simultaneously gives

\[
L_s = \frac{2(W_{ms} - W_{mm})}{I_{test}^2}
\]

\[
L_m = \frac{2W_{ms} - W_{mm}}{I_{test}^2}
\]

\[
L_{lk} = 2\frac{3W_{ms} - W_{mm}}{I_{test}^2}.
\]

Therefore for $N_t$ number of turns per phase, all the inductances are multiplied by $N_t^2$. Note that because of the nonlinearity of the steel these inductances are nonlinear and depend on the current excitation level. From the power electronics point of view these nonlinearities have negligible consequence.

### 3.5.4 Thermal Performance

Once the target motor torque specifications have been met, the temperature profile in the motor is simulated via finite element analysis. To find the temperature rise in the motor, we assume that the inner surface of the stator back-iron and the outer surface of the rotor
back-iron are at ambient temperature. We compute the total $I^2R$ losses in the motor while producing maximum torque at stall and assume that no heat is conducted longitudinally through the inner and outer periphery of the motor. In practice, there will be some heat convected away from the end turns and the exposed axial surfaces of the motor. Therefore, by ignoring these modes of heat transfer, the temperature rise obtained here is conservative. Figure 3-18(a) shows the temperature profile in the motor for 2.7 kW of total power dissipation.

To facilitate conductive heat transfer in the winding area, the windings are assumed to be vacuum impregnated with STYCAST-2850FT epoxy encapsulating resin with Catalyst 11 hardener. The thermal conductivity of the epoxy-hardener mixture is is 1.8 W/m·°C [47], and the slots are lined with 5 mils of Nomex paper, whose thermal conductivity is 1 W/m·°C. The thermal conductivity thermal conductivity of the copper is 390 W/m·°C. The effective thermal conductivity of the winding, insulation, encapsulant, and slot-liner mixture is difficult to compute analytically. In [48], Torquato and Lado computed bounds on the effective conductivity $\sigma_e$ of a composite material consisting of infinitely long conducting circular cylinders of conductivity $\sigma_2$, randomly distributed through an insulating medium of conductivity $\sigma_1$, for volume fractions of up to 65%. The effective conductivity $\sigma_e$ lies between the Milton’s fourth order lower bound $\sigma_L^{(4)}$ and the fourth order upper bound $\sigma_U^{(4)}$, where

\[
\sigma_U^{(4)} = \sigma_2 \left[ \frac{(\sigma_1 + \sigma_2)(\sigma_1 + \langle \sigma \rangle) - \phi_2 \xi_1 (\sigma_2 - \sigma_1)^2}{(\sigma_1 + \sigma_2)(\sigma_2 + \langle \sigma \rangle) - \phi_2 \xi_1 (\sigma_2 - \sigma_1)^2} \right] \tag{3.48}
\]

\[
\sigma_L^{(4)} = \sigma_1 \left[ \frac{(\sigma_1 + \sigma_2)(\sigma_2 + \langle \sigma \rangle) - \phi_1 \xi_2 (\sigma_2 - \sigma_1)^2}{(\sigma_1 + \sigma_2)(\sigma_1 + \langle \sigma \rangle) - \phi_1 \xi_2 (\sigma_2 - \sigma_1)^2} \right] \tag{3.49}
\]

where

\[
\langle \sigma \rangle = \sigma_1 \phi_1 + \sigma_2 \phi_2 \tag{3.50}
\]

\[
\langle \tilde{\sigma} \rangle = \sigma_2 \phi_1 + \sigma_1 \phi_2 \tag{3.51}
\]

and $\phi_2$ and $\phi_1$ are the volume fractions of the cylinders and host material respectively, and
Figure 3-18: Finite element simulation of temperature distribution with 2.7 kW of armature power excitation.
Figure 3-19: Variation of Milton's fourth order bounds on effective thermal conductivity with volume fraction.

are related by

$$\phi_2 = 1 - \phi_1 .$$  \hspace{1cm} (3.52)

The parameters $\xi_1$ and $\xi_2$ in Equations (3.48) and (3.49) model the cylinder-to-cylinder interactions for volume fractions up to 0.65 and are computed in [48] to be

$$\xi_2 = \frac{\phi_2}{3} - 0.05707\phi_2^2$$

$$\xi_1 = 1 - \xi_2 .$$  \hspace{1cm} (3.53) (3.54)

Figure 3-19 shows the upper and lower bounds on the effective thermal conductivity of parallel cylindrical inclusions with conductivity 390 W/m-°C embedded in a host material with conductivity 390 W/m-°C. Assuming that the volume fraction of the magnet wire insulation and the slot liner within a slot is small compared to that of the copper and epoxy, we estimate the effective thermal conductivity of the copper-epoxy mixture to be between 8.18 W/m-°C and 51.41 W/m-°C for a copper volume fraction of 0.6. For the simulation in Figure 3-18, an effective thermal conductivity of 2 W/m-°C is assumed for the whole slot, and the thermal conductivities for the remaining bulk regions are 9 W/m-°C for the
magnets, 0.033 W/m·°C for the air in the running clearance and 20 W/m·°C for the rotor and stator steels, as shown in Figure 3-18(b).

Figure 3-18(a) shows that there is a 50 °C rise within the motor and the hot-spot occurs in the airgap, close to the magnets. Assuming that the inner surface of the stator back-iron is kept at 70°C (i.e 10°C above the maximum operating ambient temperature of 60°C) while removing all the head radially, the hot spot within the motor attains a temperature of 120 °C. The winding insulation is Class H and can withstand 180°C, and the Curie temperature of Nd-Fe-B is 300 °C but we need to guard against demagnetization of the rotor magnets by the armature reaction fields. At 120°C the coercive force of the magnets is 350 kA/m, down by 65% from its value at of 1000 kA/m 20°C. At temperatures above 75°C, the “knee” of the demagnetization curve moves into the second quadrant. To prevent magnet demagnetization at 120°C the the magnetic flux density in the leading edge of the magnets must be kept below 0.4 T. From finite element analysis, we find that the flux density at the leading edge of the magnets at full load is 0.36 T. Therefore we risk demagnetization of the permanent magnets if the armature current is raised above 75 A. Operation at -40°C is not a problem for Nd-Fe-B magnets since the coercive force has a negative temperature coefficient. In short, the peak torque can be maintained indefinitely if all the 2.7 kW of heat can be removed while keeping the inner periphery of the stator back-iron at 70°C.

For operation over the Urban and Highway drive cycles, the average motor torque is about 90 Nm, a factor of one-fifth lower than the peak constant torque. Therefore the power dissipation associated the the torque-producing current is down by a factor of 25. Total hysteresis and eddy current losses when commanding this amount of torque at the maximum speed of 1300 rpm is estimated to be 300 W. Therefore, we expect a temperature rise of about 9°C within the motor. Thus under normal driving conditions, the motor can be air cooled due to the reduced losses.

If the motor is to be air cooled, then it is necessary to obtain a conservative estimate of how long the peak torque can be sustained. To obtain this estimate we assume that all the heat dissipated in the copper goes to raise the temperature of the copper and stator steel.
Thus the time, $t_{(50^\circ C)}$, to attain a $50^\circ C$ temperature rise in the motor can be computed by

$$t_{(50^\circ C)} = \frac{50 \ (M_{ss}c_{ss} + M_{cu}c_{cu})}{P_{\text{ohm}}^{\text{max}}} \ \text{mins}$$  (3.55)

where $M_{ss}$ and $M_{cu}$ is the mass in kilograms of the stator steel and copper respectively. The specific heat capacity of the stator steel, $c_{ss}$, is 502 J/kg/$^\circ$C and that of copper, $c_{cu}$, is 384 J/kg/$^\circ$C. With $M_{ss} = 11.4$ kg, $M_{cu} = 6.1$ kg, and an ohmic power dissipation of 2.7 kW, the maximum torque can be sustained for 2.5 minutes before there is a $50^\circ C$ temperature rise within the motor.
Chapter 4

Wide Speed Range Operation of PMSMs Without Flux Weakening

4.1 Introduction

Traction drives generally require operation at constant torque up to a corner speed $\omega_c$ and constant power up to a top speed $\omega_{max}$ as shown in Figure 4-1. The ratio of $\omega_{max}$ to $\omega_c$ is normally referred to as the speed range. For machines with variable rotor field excitation, such as wound-field synchronous motors, the speed at which the transition from constant torque to constant power operation is made has been traditionally referred to as the base speed. This speed normally corresponds to that at which the back-emf of the motor at full field equals the battery or dc bus voltage.

To make better utilization of the inverter’s power semiconductor devices, the field current is made inversely proportional to motor speed in the constant power region to reduce the airgap flux so as to keep the back-emf of the motor constant with speed, as also shown in Figure 4-1. The armature current is however kept constant so that the motor torque decreases in an inverse proportion to its speed to produce constant mechanical power.

Unlike the wound-field synchronous motor, where the field current can be reduced to lower the back emf of the motor for constant power operation, PMSMs operate under a fixed airgap flux. Therefore, left alone, the back-emf grows linearly with speed and consequently makes PMSMs top-speed-limited up to the dc supply voltage.
Figure 4-1: Relationships between motor torque, airgap flux, back-emf, armature current and mechanical power under flux-weakening.

To extend the speed range of PMSMs, flux weakening control has been the common practice[49, 50]. The basic idea of flux-weakening control is to use additional armature current to create a magnetic field that opposes or weakens the airgap flux so that the motor's back-emf can be kept constant. For efficient flux-weakening control, the PM rotor has to exhibit negative saliency so that direct-axis current used to weaken the airgap flux also produces reluctance torque. This requirement often results in complicated rotor designs such as the buried magnet (interior or inset) type, which have different direct and quadrature axes inductances and are often more difficult to manufacture compared to their surface-mounted counterparts. Operation under flux-weakening control, compromises motor efficiency since the direct-axis current used to weaken the airgap flux dissipates heat and produces little or no reluctance torque. In motors with square-wave winding distribution, where the back-emf of the motor is close to trapezoidal, the term “phase-advance” is used to describe
the flux weakening process. Under phase-advance operation, the inverter leg driving the commutating phase of the motor is turned on some time before the peak of the back-emf of that phase, allowing current to build up in the commutating armature winding during the rising edge of its back-emf waveform.

In addition to poor efficiency, phase advance yields only a modest increase in speed range and significant torque ripple, which is undesirable in direct-drive applications at low speeds. Thus in EV propulsion applications where system efficiency and torque ripple at low speeds are important, flux weakening or phase-advance is a sub-optimal method of achieving an extended speed range operation [53]. The problem of rotor magnet demagnetization, furthermore, becomes exacerbated during the flux-weakening mode of operation. As mentioned earlier, one of the benefits of flux-weakening is the better utilization of the Volt-Ampere (V-A) rating of the power semiconductor devices. However, if the power semiconductor devices are not rated to withstand the highest motor back-emf, in the unlikely but possible event where the microprocessor controlling the inverter undergoes a reset, large currents can be injected into the traction battery or dc power supply by the motor. This failure mode can be catastrophic in a vehicle propulsion application.

In summary, even though flux-weakening control permits savings on the V-A ratings of power semiconductors, this mode of operation complicates rotor design and fabrication, and increases torque ripple which is undesirable especially at low speed. It also lowers motor efficiency while increasing its cooling requirements, and makes the drive system prone to potentially catastrophic failures. In view of the above shortcomings of flux weakening, we seek an alternative to efficiently extend the operating speed range of a PM synchronous motor.

Figure 4-2 shows how the motor torque, airgap flux, back-emf, armature current and mechanical power vary with motor speed when flux weakening is not employed. In this mode of operation, the power semiconductor switches are rated to withstand the highest back-emf of the motor so that the catastrophic failure mode described above is avoided. The back-emf is allowed to grow linearly with motor speed and the motor's number of turns is adjusted so that the peak back-emf is below the dc bus voltage at top speed. The advantages accruing from this mode of motor operation include extended constant power speed range,
reduced torque ripple and more importantly, reduced heat removal requirements while producing constant power. This reduced thermal burden stems from the fact that backing off on the stator current for lower motor torque reduces substantially the $I^2R$ losses in the motor which is the major contributor to heat generation in the motor. Consequently, air cooling may be successfully employed in cooling the motor over the drive cycle.

Given that the power semiconductor switches are sized to withstand the highest back-emf of the motor for the reasons outlined above, the question that remains is how to choose the number of turns of the motor. Should the number of turns be chosen so that the back-emf never exceeds the battery voltage, resulting in a low-voltage/high-current drive which allows the use of a standard three-phase voltage-source inverter or are there some benefits in going to a high-voltage/low-current drive? We show shortly that there are substantial gains in inverter efficiency by employing a high-voltage/low-current drive, in addition to
Figure 4-3: Power circuit of a standard 3-phase IGBT voltage source inverter.

reduced snubbing and switching current filtration requirements.

4.2 Standard 3-Phase Voltage Source Inverter

Figure 4-3 shows the power circuit for a standard 3-phase IGBT voltage source inverter. Three two-quadrant half-bridges are used to chop the battery voltage to match the motor back-emf as well steer currents in the appropriate phase windings to ensure smooth and continuous motor rotation. Two-quadrant operation of the inverter is necessary for regenerative braking, where the traction motors are used as generators to capture the kinetic energy of the vehicle.

Consider a 15 kW (20 hp) traction drive that operates from a nominal battery voltage of 250 V. Allowing about 50 V of headroom for commutation at top speed, the inverter has to supply a dc current of 300 A at the corner speed where the motor back-emf is 50 V, in order to achieve a 4:1 constant-power speed range. In other words, the inverter has to process the total 15 kW plus its power conversion and motor losses at 50 V. Current state-of-the-art IGBT modules are available in two voltage ratings, namely 600 V and 1200
V. With a forward voltage drop of 2.7 V, conduction losses in the IGBT modules amount to 1.62 kW since there are always two devices in the conduction path of the topology in Figure 4-3. Neglecting all other losses, conduction losses alone constitute an efficiency drop of 9.7%. Associated with this drop in power conversion efficiency is the problem of heat removal from the devices. If the back-emf of the motor were increased to say 200 V at the corner speed, 1200 V IGBTs will be needed in order to be able to withstand the motor back-emf at top speed, and the conduction losses can be cut down by a factor of four since the on-state voltage for 1200 V, low-current IGBT modules is essentially the same as that for 600 V high-current modules [20].

Another problem with the low-voltage/high-current inverter is the large rms current that the bus capacitors must filter. Consider the buck converter shown in Figure 4-4(a), which represents one leg of the standard 3-phase inverter shown in Figure 4-3. For the sake of
Figure 4-5: RMS current through bus capacitors over the entire region of operation of the motor.

discussion, the motor inductance $L_m$ is assumed to be infinite so that the motor current is essentially dc. The armature resistance is assumed to be zero so that ohmic losses in the motor can be ignored. With a 300 A motor current, the dc battery current is 60 A, assuming perfect conversion efficiency. The resulting duty cycle is 20%. Thus, during 20% of the time when the motor is directly connected to the dc bus, $C_B$ sources 240 A of current as shown in Figure 4-4(b). During the remaining 80% of the time when the motor current is freewheeling through the low-side diodes, $C_B$ is charged by the 60 A of battery current. The net ac current that $C_B$ has to filter at this operating point is 120 A rms. Figure 4-5 shows the rms current through $C_B$ over the entire region of operation of the motor using the methodology above. Note that, in general, the rms ripple current through $C_B$ is zero at both zero and unity duty cycles and the maximum value occurs at 0.5 duty cycle. The rms current is also proportional to the motor current. For the standard inverter under discussion, the maximum rms ripple current through $C_B$ occurs at the “corner point” (as shown in Figure 4-5) since operation at 0.5 duty cycle, for this inverter, occurs in the constant power mode of operation at which point the motor current is down by a factor of
Figure 4-6: Resistor-Capacitor-Diode snubber for high-current IGBTs.

2.5 from its constant torque value. This peak current filtration requirement is harsh on the bus capacitors which are often electrolytic and are limited in their ripple current carrying capability, and hence require several capacitors to be paralleled even though the additional capacitance is not necessary from an impedance point of view.

While switching large currents, the IGBTs have to be adequately snubbed to keep them in the safe operating region. The voltage overshoots that occur in switching transients is proportional to the amount of current being switched. For proper device operation, the recommended snubber for a 600-V 300-A IGBT require the complicated resistor-capacitor-diode (RCD) combination shown in Figure 4-6, in conjunction with intricate low-impedance bus designs to minimize the increased current-related voltage overshoots associated with switching large currents [20].

In addition to the above drawbacks of a low-voltage/high-current design from the power-electronics point of view, the motor that matches the low-voltage inverter requires fewer turns, which necessitates that the windings be made of large conductors. Larger gauge wires are difficult to wind and their end turns are difficult to control. Larger slot openings are
required, and if the parallel conductors are used, the wires have to be properly transposed to minimize circulating currents. To address the above concerns, we explore the use of an unconventional inverter design that permits motor operation at high voltage.

4.3 Novel 3-Phase Voltage Source Inverter

Figure 4-7 shows a modification of the standard 3-phase voltage source inverter to include a bi-directional boost converter in the dc-link. This modification makes possible efficient wide speed range operation of PMSMs without flux weakening while circumventing the problems associated with the standard voltage source inverter discussed in the previous section. In this inverter topology, a constant power bi-directional boost converter is used to step up the traction battery voltage as necessary to match the motor back-emf [51, 52].

The operation of the inverter is as follows. When the back-emf of the motor is below the battery voltage, the boost converter is disabled by turning off $S_1$ and $S_2$ and the inverter operates as a standard voltage source inverter where the switches $S_3$ -- $S_8$ are used to chop the battery voltage as well as commutate the motor. When the back-emf exceeds the battery voltage, the boost converter is used to step up the battery voltage to match the motor back
emf and the switches $S_3 - S_8$ are only used to commutate the motor. This way, we minimize
additional switching losses associated with high-frequency pulse-width modulation (PWM).
The regions under the torque-speed curve where the inverter operates in the buck and boost
modes are schematically shown in Figure 4-8. The slope to the buck/boost boundary is due
to the voltage that develops across the non-zero motor resistance.

To implement regenerative braking when the motor back-emf is below the battery
voltage, $S_3 - S_8$ are used to boost the motor back emf to match the battery voltage, thus
capturing the motor regenerative energy and storing it in the battery. In this mode of
operation, $S_2$ is turned on while $S_1$ remains off. When the back-emf is greater than the
battery voltage, then the switches $S_3 - S_8$ are used to rectify the motor's back-emf and down-
converted to match the battery voltage by pulse-width modulating $S_2$. In the regenerative
braking mode of operation, mechanical energy from the vehicle is converted to electric
energy and stored in the traction battery or a high-power intermediate energy storage device
such as a flywheel.

In the remaining sections of this chapter, we compare the relative merits and shortcom-
ings of the two inverter topologies in terms of energy storage component requirements,
power conversion efficiency, installed V-A capacity, snubbing requirements, cost, and control complexity.

4.4 Energy Storage Requirements

The only energy storage components required in the standard inverter topology of Figure 4-3 are the capacitors across the traction battery, which are required to filter the ripple currents generated by the switching of the three half-bridge inverter switches. As we have shown earlier, these bus capacitors have to filter 120 A rms at the “corner point”. For operation with the high-voltage inverter shown in Figure 4-7, the number of turns of the motor is adjusted by a factor of four to yield a back-emf of 200 V at the “corner point”. Remember that for the low-voltage inverter, we allowed a margin of 50 V above the back-emf to effect commutation. To achieve the same rate of current buildup in the commutating winding as before, we now require 200 V of voltage margin above the back-emf since the inductance increases as the number of turns squared, and the commutation current is inversely proportional to the number of turns. Thus for a back-emf of 800 V at top speed and an additional 200 V for commutation, we have a headroom of 200 V to accommodate current related voltage overshoots and switching voltage ripple. As mentioned earlier the ripple current in the bus capacitors is a maximum when switching the maximum current at 50% duty cycle.

Using the ripple current computation methodology illustrated in Figure 4-4, we calculate the rms current through the bus capacitors when the inverter is operating in the buck mode, and the result is shown in Figure 4-9. With the increase in the number of turns by a factor of four, the torque constant increases by the same factor and the maximum constant torque current is now 75 A dc. The resulting rms current ripple that must be filtered by the bus capacitors is only 37.5 A, a factor of 3.2 reduction. The maximum rms current in the boost mode is lower than that of the buck mode due to the current smoothing action of the boost inductor.

At a switching frequency of 20 kHz and a traction battery internal resistance of about 0.3 Ω, 265 μF of capacitance constitutes an impedance that is one-tenth that of the battery,
Figure 4-9: RMS current through bus capacitors in the buck mode of operation.

thus constituting an adequate low-impedance path for current filtering. This amount of capacitance is readily obtainable in a single electrolytic capacitor, but not enough current filtering capability due to the high dissipation factor of electrolytics. Therefore, the number of capacitors needed is often proportional to the rms current. In view of the rms ripple current that must be filtered, the novel inverter requires less than a third of the number of bus capacitors needed for the standard voltage source inverter.

The novel inverter of Figure 4-7 requires two additional energy storage elements; a front end inductor $L_u$ and an output capacitor $C_{DC}$ for the bi-directional boost converter. The size of an inductor is, in general, proportional to the amount of energy the inductor must store. Therefore we determine the boost inductor parameters by minimizing the amount of energy that it must store. The peak energy $E_{L_u}$ stored in an inductor of inductance $L_u$, carrying a peak current $I_{pk}$ is given by

$$E_{L_u} = \frac{1}{2} L_u I_{pk}^2 .$$ (4.1)

For a constant power boost converter with an input voltage $V_{in}$ and an output voltage $V_{out}$
switching at a constant frequency $f_{sw}$, the energy stored in the boost inductor is minimized when the converter is operating at the edge of discontinuous conduction. The critical inductance $L_{u,\text{crit}}$ that minimizes the energy storage is given by

$$L_{u,\text{crit}} = \frac{V_{\text{in}}^2}{2P_{\text{out}}f_{\text{sw}}} \left[1 - \frac{V_{\text{in}}}{V_{\text{out}}} \right]$$  \hspace{1cm} (4.2)

where $P_{\text{out}}$ is the power processed. At the edge of discontinuous conduction, the peak inductor current is equal to twice the average input current $I_{\text{in}}$. The corresponding minimum magnetic energy stored is therefore given by

$$E_{L_{u,\text{min}}} = \frac{P_{\text{out}}}{f_{\text{sw}}} \left[1 - \frac{V_{\text{in}}}{V_{\text{out}}} \right].$$  \hspace{1cm} (4.3)

For $V_{\text{in}} = 250$ V, $V_{\text{out}} = 1000$ V, $f_{\text{sw}} = 20$ kHz, and $P_{\text{out}} = 15$ kW, the critical inductance $L_{u,\text{crit}} = 78 \ \mu$H and the required magnetic energy stored by the boost inductor is 0.56 J. The output capacitor $C_{\text{DC}}$ is chosen to be 40 $\mu$F to yield an output ripple voltage of 25 V peak-to-peak. The electric energy storage $E_C$ is given by

$$E_C = \frac{1}{2} C_{\text{DC}} V_{p_{\text{k}}}^2 = 25 \text{J}$$  \hspace{1cm} (4.4)

for $V_{p_{\text{k}}}$ of 1025 V. Thus the new inverter requires about 0.56 J of additional magnetic energy storage and 25 J of high-voltage electric energy storage.

### 4.5 Conversion Efficiency

By moving the “corner point” of the inverter to 200 V, the conduction losses go down by a factor of four since the current is reduced by one-fourth. However, since we have one extra device conducting in the boost converter in the new topology, the total conduction losses is only lowered by a factor of 2.67, from 1.62 kW to 606 Watts. With our switching strategy of pulse-width modulating only one section of the converter at a time, the switching losses to first order are the same for both converters, ignoring the switching losses associated with the low-frequency six-step commutation when the boost converter is in operation. Additional
energy is lost in the boost inductor but will be ignored in this qualitative comparison, since the exact loss is not known until the inductor has been designed.

4.6 Installed Volt-Ampere Capacity

The standard inverter requires six 600-V, 300-A two-quadrant switches, making the installed V-A capacity 1.08 MVA, whereas the modified inverter requires eight 1200-V, 75-A two quadrant switches, resulting in 0.72 MVA. This corresponds to a factor of 1.5 reduction in installed V-A capacity for the new inverter.

4.7 Snubbing Requirements

The reduced switching current requirement in the new high-voltage inverter, in conjunction with a low-inductance bus structure, allows the use of the simple snubber in shown in Figure 4-10, where low-inductance high-voltage polypropylene snubber capacitors are connected across the collector and emitter of the half-bridge as opposed to the complicated RCD snubber shown in Figure 4-6.

4.8 Component Cost

The total cost of the power circuit in the standard inverter is effectively a weighted sum of the installed V-A capacity of the power semiconductors and energy storage in the bus capacitors, and the cost of snubber components. We have shown in the previous sections that by using the new inverter topology, the cost of the electrolytic bus capacitors is reduced by a factor of 3.2 and we save on the power semiconductor V-A ratings by a factor of 1.5. Furthermore, the snubber for the standard inverter is more costly than that in the new inverter due to the added diodes. It is anticipated that, in general, the cost savings deriving from the reduced energy storage in the bus capacitors, reduced power semiconductor V-A ratings, and simplified snubber will more than compensate for the added cost of a boost inductor and boost output capacitors and two additional gate drives.
4.9 Control Complexity

Torque control in the standard 3-phase voltage source inverter is easily accomplished with as simple proportional plus integral (PI) digital controller. In the new inverter, however, in addition to a PI controller for the buck section of the inverter, we require additional controllers for the boost section which operates mostly in discontinuous mode. These different controllers in the new inverter have to relinquish control to each other as needed, in response to the torque demands of the motor. Although the control algorithm for the new inverter is more complex than the standard inverter, it can be implemented at no additional cost with a microcontroller.

4.10 Conclusions

In applications where energy efficiency is at a premium, the use of power electronics instead of flux-weakening to extend the speed range of PMSMs offers significant advantages in motor efficiency and cooling requirements, while avoiding potentially catastrophic failure
modes associated with inverter reset during operation in the flux-weakening mode. By using the unconventional high-voltage inverter described in this chapter, the efficiency of the inverter is improved due to lower semiconductor conduction losses. The installed V-A capacity is reduced as well as the snubbing and current filtration requirements of the dc bus capacitors. Total semiconductor and energy storage component cost is about the same, if not less, for the new inverter compared to the standard inverter.
Chapter 5

Prototype Wheel Motor: Power Electronics and Control

5.1 Introduction

In this chapter, we design the prototype inverter that will be used to drive the wheel motor designed in Chapter 3. We start by specifying component values for the power circuit and then proceed to derive averaged plant models to be used for control. We then design digital controllers around these plants and implement these controllers with an MC68HC16Z1 microcontroller. For our computations in this chapter, the traction battery will be modeled as a 336-V voltage source in series with a 0.3 Ω series resistance[42].

5.2 Power Circuit

The power circuit of the of the novel inverter is reproduced in Figure 5-1 for ease of reference. In this section, we discuss the design of the experimental inverter to be used with the prototype wheel motor designed in Chapter 3, with regard to semiconductor and energy storage component specification and compute the theoretical efficiency of the inverter. Our original intent was to switch the inverter at 20 kHz to avoid audible noise, but this PWM frequency was not readily obtainable from the MC68HC16Z1 microcontroller used for control. Therefore the inverter was switched at 16.44 kHz, which is the closest
programmable frequency to 20 kHz.

5.2.1 Semiconductor Component Specification

As discussed in Chapter 4, to achieve efficient wide speed range operation without flux weakening, the semiconductor switches must be sized to withstand the highest back-emf of the motor and have adequate headroom for the extra voltage needed for commutation. In addition, one must also ensure that the switching-related voltage ripple on the output capacitors of the boost converter does not cause the voltage rating of the IGBTs to be exceeded. With a back-emf constant of 6 V/rad-s, the expected maximum back-emf at 1300 rpm is 800 V. Budgeting 200 V for commutation, 1200 V IGBTs will be used. The maximum constant torque required sets the current rating of the IGBTs. Since 75 A is required to produce 450 Nm of torque, we select 100 A devices to allow a margin for safety. To simplify the layout of the power circuit and gate-drive circuitry we use half-bridge intelligent IGBT modules from Powerex which have integral gate-drive and protection circuitry.
5.2.2 Energy Storage Component Specification

A. Bus Capacitors

As shown in Chapter 4, the maximum rms current that the bus capacitors must filter is 38 A. From an impedance point of view, 322 \( \mu \)F constitutes an impedance of 0.03 \( \Omega \) (one-tenth the battery impedance) at the switching frequency of 16.44 kHz. Power electrolytic capacitors have high dissipation and are therefore limited in the ripple current they can filter. For example, a 2,400 \( \mu \)F, 450 VDC high-ripple electrolytic capacitor can only handle 9.4 A rms of ripple current at 20 kHz\(^1\) [54]. The ripple current carrying capability can only be increased by increasing the capacitance. Therefore, in general, several capacitors must be paralleled for high ripple current filtration. For the prototype inverter, we parallel six Mallory CGH242T450W4L capacitors to yield a ripple current capability of 56.4 A. Notice that the resulting capacitance 14.4 mF is very much in excess of what is needed from an impedance point of view.

B. Boost Inductor

As shown in Chapter 4, the inductance \( L_{u,\text{crit}} \) that minimizes the magnetic energy storage is given by

\[
L_{u,\text{crit}} = \frac{V_{\text{in}}^2}{2P_{\text{out}} f_{\text{sw}}} \left[ 1 - \frac{V_{\text{in}}}{V_{\text{out}}} \right].
\]

(5.1)

With a maximum output voltage of 1050 V (800 V from the back-emf, 200 V for commutation and 50 V for Ohmic drops and current related overshoots), a low battery voltage of 250 V [42], and a maximum output power of 15 kW while switching at 16.44 kHz, the inductance required is 97 \( \mu \)H. The corresponding inductor was constructed by winding 48 turns of AWG 12 Litz wire (259 strands of AWG 36) on a 4-inch outer diameter, Hi-Flux\(^T\)\(^M\) powder (HF) toroidal core\(^2\) [56]. The resulting copper and core losses at full power is 106 W and 170 W respectively.

HF cores are composed of 50% nickel and 50% iron, and have a higher saturation flux density (1.5 T) compared to Molybdenum Permalloy Powder (MPP) or ferrites which have

\(^1\)Mallory Part no. CGH242T450W4L
\(^2\)Arnold Engineering Part no. HF-400060-2
saturation flux densities of 0.7 T and 0.3-0.5 T, respectively. The high saturation flux of HF cores make them suitable for applications requiring high magnetic energy storage, yielding significant reduction in inductor size and cost compared to ferrites or MPP. The permeability of HF cores is also less sensitive to changes in ac excitation, dc bias and temperature.

C. Boost Output Capacitors and Snubbers

The function of the capacitors at the output of the boost converter is to hold the output voltage while the boost inductor is being charged. These capacitors must filter significant ripple current as charge is dumped into them periodically from the boost inductor. Therefore these capacitors must have a low dissipation factor in order to prevent excessive heating. Both polypropylene capacitors or low dissipation electrolytic capacitors may be used for this application. However, the required voltage rating of 1200 V precludes the use of single electrolytics since 1200 V is well above the voltage rating of commercially available electrolytics. Furthermore, as we saw earlier, attaining a reasonable ripple current handling capability requires a large capacitance. Excessive capacitance at the output of the boost converter is undesirable since it limits our ability to dynamically shape the output voltage to control the motor current in response to rapid changes in torque demand.

With 100 $\mu$H of boost inductance, a capacitance value of 40 $\mu$F is chosen as a reasonable balance between output ripple voltage and capacitor cost. With this amount of output capacitance, the maximum ripple voltage across the output capacitor is computed to be 25 V and the rms ripple current is 30 A. Due to difficulty in procuring a standard capacitor with these voltage and current ratings, the 40 $\mu$F output capacitor is implemented with four 10 $\mu$F, 1500 V capacitors\(^3\). Each of the 10 $\mu$F capacitors procured is capable of filtering 207 A of ripple current, thus the capacitors are extremely oversized for the current handling capability required [55]. Furthermore, the size of a capacitor is proportional to the energy storage, which is proportional to the square of the voltage. Thus, rated at 1500 V, these capacitors are a factor of 1.56 larger than necessary, but no appropriate capacitor was readily available.

\(^3\)GE Part no. 97F8625
To minimize current-related voltage overshoot during switching transients, 2-μF low-inductance polypropylene snubber capacitors are connected directly across the collector and emitter of the IGBT half-bridge modules.

### 5.2.3 Semiconductor Losses

The inverter in Figure 5-1 is effectively a cascade of hard-switched boost and buck converters, and Figure 5-2 shows the regions of operation of the buck and boost converters. As mentioned earlier, the buck converter is used to drive the motor until its terminal voltage (the sum of the back-emf and the resistive voltage drop) becomes equal to the inverter input voltage, at which point the boost converter takes over.

Figure 5-3(a) shows the magnitude of the dc input current input current into the inverter in its entire region of operation. This plot is generated with Maple V (source code in Appendix D.3) under the assumption that the inverter is 100 % efficient, and all four wheel motors are operating simultaneously at the same point underneath the torque-speed...
Figure 5-3: Variation of inverter's input current and input voltage with motor current and back-emf.

Under these circumstances, the dc current into the inverter $I_{in}$ can be written as

$$I_{in} = \frac{V_{B,(oc)} - \sqrt{V_{B,(oc)}^2 - 16R_{int}V_{out}I_{out}}}{8R_{int}} \quad (5.2)$$

where $I_{out}$ is the inverter output dc current, $V_{out}$ is the inverter output voltage, $V_{B,(oc)}$ is the battery open circuit voltage and $R_{int}$ is the battery internal resistance. The output voltage $V_{out}$ is related to the motor resistance $R_m$ and the back-emf $V_{emf}$ by

$$V_{out} = V_{emf} + I_{out}R_m \quad (5.3)$$
Figure 5-3(b) shows the corresponding input voltage \( V_{in} \) to the inverter. The input voltage is the battery open-circuit voltage less the voltage drop across its internal resistance, and is given by the equation

\[
V_{in} = V_{B_{e}(ac)} - 4I_{in}R_{int}.
\]  

(5.4)

The slope to the line in Figure 5-2 that forms the boundary between the buck and boost regions of operation of the inverter is due to the drop in the input voltage with increased motor current and to a small extent the resistive drop across the motor that increases with increased motor current.

Hard-switched converters incur switching losses at both the turn-on and turn-off transitions. Figure 5-4 shows typical switching waveforms for a half-bridge intelligent module and an illustration of the associated power loss. The energy lost during switching transitions, which correspond to the shaded areas in Figure 5-4(b), is proportional to the voltage and current being switched and the duration of the switching interval. The average power loss is therefore the energy loss per switching cycle multiplied by the switching frequency. Graphical data on the energy dissipated during both turn-on and turn-off transitions including diode reverse recovery losses are available from the power semiconductor manufacturer's data sheets [20].

Figure 5-5(a) shows the variation of switching losses with the motor's back-emf at various motor currents. This figure is generated with Maple V subroutines in Appendix D.3 using the graphical data on switching and conduction losses provided in [20]. Note that the back-emf is proportional to speed and the motor current is proportional to torque, thus the Figure can be interpreted as representing the switching losses underneath the torque-speed envelope. Our switching approach of not switching both converters simultaneously help to reduce the total switching losses. In the buck mode of operation, the switching losses for a given current is independent of motor speed since the inverter is switched at a constant PWM frequency from a fixed bus voltage. In the boost mode however, even though the boost converter switches at a constant PWM frequency, the output voltage of the boost converter is proportional to the motor speed and hence we expect the switching losses to increase with the increased output voltage. Furthermore, in this mode of operation, the three
inverter legs are operated in a six-step fashion to commutate the motor. Not only is the commutation frequency proportional to the motor speed, the switching losses deriving from this section of the inverter is also proportional to the output voltage. These two phenomena explain the increase of the switching losses with the motor speed as shown in Figure 5-5(a). Note that the maximum commutation frequency is 1300 Hz at top speed, thus the maximum switching losses from the six-step commutation is less than one-tenth of the losses from the high frequency PWM section.

Due to the non-zero collector-to-emitter voltages of the IGBTs, and non-zero forward
voltages of the anti-parallel diodes, power is dissipated in the IGBT module when conducting current. Graphical data on the on-state voltage drop across the power device are also provided in the power semiconductor manufacturer's data sheets [20]. The conduction power loss in a device is the time-average of the integral of the product of the voltage across and current through the device. Figure 5-5(b) shows the conduction losses in the inverter. The conduction losses are highest in the buck regime of operation for two reasons: (i) the highest currents are switched in this constant torque regime of operation, and (ii) there is an
extra diode from the disabled boost converter in the power flow path from input to output. The sharp drop in conduction losses in traversing the buck/boost operating boundary is due to the elimination of an extra junction drop form the power flow path.

Figure 5-5(c) shows the total semiconductor losses, which is the sum of the switching and conduction losses. From the figure we see that the maximum semiconductor loss is about 750 W and it occurs at the "corner-point" of the inverter.

5.2.4 Inverter Efficiency

The efficiency of the inverter $\eta_{inv}$ is computed as

$$\eta_{inv} = \frac{V_{out}I_{out}}{V_{out}I_{out} + P_{semi} + P_{ind} + P_{gd}} \tag{5.5}$$

where $P_{semi}$ is the sum of the switching and conduction losses of the power semiconductors, $P_{ind}$ is the copper and core loss in the boost inductor, and $P_{gd}$ is the power loss in the gate drives and control electronics, assumed to be 20 W. Figure 5-6 shows the theoretical efficiency map of the inverter as computed with Maple V in Appendix D.3. The medium torque, medium speed region attains efficiencies in excess of 96%. The weighted efficiencies
over the Urban and Highway drive schedules are 94 % and 95 %, respectively.

5.3 Control

The inverter in Figure 5-1 has four modes of operation. The first mode of operation is the “buck-motoring” mode, when the motor is motoring with a back-emf below the battery voltage. The second mode of operation is the “boost-motoring” mode, when the motor is motoring with a back-emf is above the battery voltage. The third mode of operation is the “boost-regenerative” mode, when the motor is regenerating with a back-emf below the battery voltage, and the fourth mode of operation is the “buck-regenerative” mode, when the motor is regenerating with a back-emf above the battery voltage. In all operating modes, we are interested in closing a feedback loop around the output current of the inverter, which is the motor current when motoring, and the battery charging current when regenerating.

In this thesis, control loops will be implemented for the first two modes of operation where electrical power is converted to mechanical power via the motor and transferred to an inertial load, as operation in these two regimes is enough to characterize the torque production capabilities and efficiency of the motor.

5.3.1 Average Models

In order to close the current feedback loop, we must develop average models that describe how the “plant”, which consists of the power circuit of the inverter and the electrical section of the motor, respond to the control variable of our choice in an average sense. In the “buck-motoring” mode of operation, the control variable of choice is the duty cycle of the buck converter since the transfer function from control to output is contains only a single pole.

Figure 5-7(a) shows an averaged circuit model of the plant in the “buck-motoring” mode of operation. The average motor current $\bar{i}_m$ is related to the duty cycle $\bar{d}$ via the equation

$$\bar{i}_m = \frac{\bar{d} V_{in} - V_{emf}}{sL_m + R_m}$$  \hspace{1cm} (5.6)
where $L_m$ is the synchronous inductance of the motor and $R_m$ is the series resistance of two motor phases, a long as $s$ is well below the switching frequency.

Figure 5-7(b) shows an averaged circuit model of the plant in the "boost-motoring" mode of operation. In this mode of operation the converter operates mostly in discontinuous conduction. Peak current-mode control is employed to suppress the dynamics associated with the boost inductor. We can therefore model the response of the motor current to the average injected current $\bar{i}_{inj}$ into the LCR tank circuit consisting of the boost output capacitors $C_{dc}$, the synchronous inductance of the motor $L_m$, and the phase to phase motor resistance $R_m$ by

$$\bar{i}_m = \frac{1}{L_mC_{dc}s^2 + R_mC_{dc}s + 1} \bar{i}_{inj} - \frac{sC_{dc}}{L_mC_{dc}s^2 + R_mC_{dc}s + 1} V_{emf} \quad (5.7)$$

Note that, unlike the buck operating mode, the plant has second-order dynamics which, in general, is more difficult to control compared to a first-order plant. The average motor current $\bar{i}_m$ is affected by the rate of change of the motor back-emf. Since we control $\bar{i}_m$ through $\bar{i}_{inj}$, the back-emf appears as a disturbance, the effect of which must be corrected by the feedback control system.

In discontinuous mode operation, the average injected current $\bar{i}_{inj}$ is related to the boost
inductor peak current $I_{pk}$ by

$$
\tilde{i}_{inj} = \frac{L_u}{2T(V_{out} - V_{in})} \cdot I_{pk}^2
$$

(5.8)

Thus

$$
I_{pk} = \sqrt{\frac{2T}{L_u}}(V_{out} - V_{in})\tilde{i}_{inj}
$$

(5.9)

and therefore to keep the loop linear, the microcontroller has to compute the proper peak inductor current from $\tilde{i}_{inj}$ as shown in Equation (5.9).

### 5.3.2 Digital Controllers

The motor current control loops are implemented digitally with the MC68HC16Z1 microcontroller. The bandwidth of the vehicle controller which interprets driver commands and issues a torque command to the wheel motor controller is 20 Hz [21]. Therefore the bandwidth of the wheel motor torque controller has to be an order of magnitude above that of the vehicle controller to allow the two controllers to be designed independently. Thus, we aim to achieve closed loop bandwidths of 200 Hz for our motor current control loops.

Figure 5-8(a) shows the general block diagram of the discrete-time torque control system and Figure 5-8(b) shows the sequencing of events during a control interval. On the kth control cycle, the (k-1)th output of the controller is applied to the plant and the motor current, boost converter output voltage, and inverter input voltage are sampled and digitized. The digitized motor current is subtracted from the commanded current to generate a current error signal. This error signal is passed through a proportional plus integral (PI) compensator, the integral part being necessary to ensure zero steady-state error. The result is then added to a scaled version of the (k-1)th controller output to be applied to the plant at the next control cycle. When the motor is operating in the regime where the boost converter is active, the boost converter output voltage is also used in computing the output of the controller, as shown by the dashed line in Figure 5-8(a).

To compute the feedback gains necessary to yield our desired response, we need to represent our control system in discrete time state-space form. The state equations for
(a) Block diagram of controller. (Dashed line indicate voltage feedback for boost converter)

(b) Sequence of control events

Figure 5-8: General block diagram of buck and boost control loops.

Continuous time linear time-invariant (LTI) system can be written in matrix form as

\[ x(t) = Ax(t) + Bu(t) \]  \hspace{1cm} (5.10)

Where \( A \) is the dynamic-coefficient or system matrix, \( x(t) \) is the state vector, \( B \) is the input-coupling matrix and \( u(t) \) is the input vector. Equation (5.10) can be represented in discrete time as

\[ x[k+1] = Fx[k] + Gu[k] \]  \hspace{1cm} (5.11)

Where

\[ F = e^{AT_s} \]  \hspace{1cm} (5.12)

\[ G = A^{-1}(e^{AT_s} - I)B \]  \hspace{1cm} (5.13)

And \( T_s \) is the sampling period.

For the buck mode of operation, the continuous-time state-space equation for the plant
is given by

\[ \frac{d}{dt} \tilde{v}_m = \frac{R_m}{L_m} + \frac{1}{L_m}(\tilde{d}V_{in} - V_{em}) \tag{5.14} \]

where the A and B matrices are scalars given by \(-R_m/L_m\) and \(1/L_m\) respectively. The discrete-time dynamic system that incorporates the integrator and the one-sample delay shown in Figure 5-8(a) is given by

\[
\begin{bmatrix}
\tilde{v}_m[k + 1] \\
z[k + 1] \\
u[k + 1]
\end{bmatrix}
= \begin{bmatrix}
F & 0 & G \\
1 & 1 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\tilde{v}_m[k] \\
z[k] \\
u[k]
\end{bmatrix}
+ \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} q[k] \tag{5.15}
\]

where the F and G matrices are defined as in (5.12) and (5.13). In Equation 5.15, the state variables \(z\) and \(u\) represent the output of the integrator and and one-sample delay element and the input \(q[k]\) represents the controller’s output in the kth control interval.

For the boost mode of operation, the continuous-time, state-space equation for the plant (assuming a constant back-emf) is given by

\[
\frac{d}{dt} \begin{bmatrix}
\tilde{v}_{dc}(t) \\
\tilde{v}_m(t)
\end{bmatrix}
= \begin{bmatrix}
0 & -\frac{1}{C_{dc}} \\
\frac{1}{L_m} & -\frac{R_m}{L_m}
\end{bmatrix}
\begin{bmatrix}
\tilde{v}_{dc}(t) \\
\tilde{v}_m(t)
\end{bmatrix}
+ \begin{bmatrix}
\frac{1}{C_{dc}} \\
0
\end{bmatrix} \tilde{v}_{inj}(t) \tag{5.16}
\]

In this case, the corresponding discrete-time dynamic system that incorporates the integrator and the delay shown in Figure 5-8(a) is given by

\[
\begin{bmatrix}
v_{dc}[k + 1] \\
\tilde{v}_m[k + 1] \\
z[k + 1] \\
u[k + 1]
\end{bmatrix}
= \begin{bmatrix}
F_{11} & F_{12} & 0 \\
F_{21} & F_{22} & 0 \\
0 & 1 & 1 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
G_{11} \\
G_{21} \\
0 \\
0
\end{bmatrix}
\begin{bmatrix}
v_{dc}[k] \\
\tilde{v}_m[k] \\
z[k] \\
u[k]
\end{bmatrix}
+ \begin{bmatrix}
0 \\
0 \\
0 \\
1
\end{bmatrix} q[k] \tag{5.17}
\]

where the F and G matrices are again as defined in (5.12) and (5.13).

Given a row vector \(P\) which contains the desired pole locations of the closed loop system, the M and N matrices in (5.15) and (5.17) are used to compute the state feedback.
matrix $K$ such that the eigenvalues of $M - NK$ are those specified in vector $P$. The elements of $K$ are the feedback gains $k_i$, $k_z$, $k_u$, and $k_v$ when operating in the boost mode. Given the state-transition matrix $M$, the input vector $N$, and the desired pole location vector $P$, the "place" routine in the Matlab\textsuperscript{4} control toolbox is used to compute the state feedback matrix $K$.

Figure 5-9 shows the relationship between the pole locations in the $z$-plane and the natural frequency $\omega_n$ and the damping ratio $\zeta$ for a standard all-pole, second-order system [57]. Even though our two control systems are of higher order (third order for the buck and fourth order for the boost) we will approximate their response to be second order by placing the poles such that there is a dominant complex pole-pair. Unlike the standard all-pole second-order system, ours contains transmission zeros. Thus for a given damping ratio $\zeta$, our system response will exhibit a higher overshoot than a all-pole second order

---

\textsuperscript{4}Matlab is a registered trademark of The Mathworks, Inc., 24 Prime Park Way, Natick, MA 01760
system. The settling time $T_{st}$ to a step input can be approximated as

$$T_{st} = \frac{4}{\zeta \omega_n}.$$  (5.18)

For our control loops, we seek a settling time 5 ms or less. Thus for a sampling period of 244 $\mu$s we seek to place our closed loop poles in the general area demarcated by the dotted line in Figure 5-9. Once initial feedback gains are obtained from Matlab, the control loops are simulated and fine-tuned with Simulink\(^5\). Simulations of the response of the two control systems are given in Chapter 6.

5.4 Experimental Circuit Implementation

Figure 5-10 shows a schematic diagram of how the inverter power circuit, microcontroller and user interface are interconnected. Desired motor torque and motor direction commands are input via a potentiometer and a keypad. The microcontroller interprets user input commands and controls the switching of the IGBTs $S_1 - S_8$ to meet user demands. The battery voltage, boost converter output voltage, boost inductor current, and two of the motor phase currents are sensed and used for control. A shaft encoder is implemented via three Hall-effect sensors, the signals from which are not only used for commutation but also for motor speed calculations. A 16-character by 4-line Optrex Liquid Crystal Display (LCD)\(^6\) is used to display relevant input and output data which may consist of commanded motor torque, motor speed, battery voltage and other parameters of interest.

5.4.1 Commutation circuits

To determine rotor magnet position relative to the stator windings, we use SS461A latching, open collector, digital Hall sensors by Micro Switch [58]. These devices have an operating temperature range of -40°C to +150°C, and operate from a dc supply voltage of 3.8 - 24 V. The maximum operate and minimum release points are 100 Gauss and -100

---

\(^5\)Simulink is a registered trademark of The Mathworks, Inc., 24 Prime Park Way, Natick, MA 01760
\(^6\)Optrex Corporation part no. DMC16433
Figure 5-10: Schematic diagram of inverter implementation.
Gauss respectively\(^7\). Figure 5-11 shows the circuitry that interface the Hall sensors to the microcontroller. The three Hall sensors are mounted 120° on the stator to detect the rotor magnet’s fringing fields, and the resulting 3-phase digital output signals are transported to the input-capture port of the General Purpose Timer (GPT) of the microcontroller. The Hall signals are phase-locked in software within the microcontroller and are used to compute motor RPM. The RPM computed is used to estimate the motor’s back-emf constant as well as adjust the user-commanded torque accordingly when it falls outside the constant power boundary. The phase-locked signals, which can be phase advanced in software to better align the Hall signals to the motor back-emf waveforms, are output through the output-compare pins of the GPT and used as inputs to combinational logic used to commutate the motor.

\(^7\)Operate and release points are strength of the magnetic fields needed to turn on and off the open collector transistor respectively
Figure 5-12: Truth table for motor commutation and phase selection.
Figure 5-13: Gate-drive interface circuitry.

Figure 5-12 is a truth-table that shows how the IGBTs are switched depending on the mode of operation of the inverter. When the INH signal is asserted the drive is disengaged and the motor coasts. The FORW signal sets the direction of motor rotation to forward. The BUCK signal indicate that the motor back-emf is below the battery voltage and the MOT signal indicate that the motor is motoring as opposed to regeneration. The signals HEA, HEB, and HEC are Hall-effect signals for phases A, B, and C respectively. The SELA, SELB, SELC signals indicate the phase current to be used for the torque feedback loop, and NEG indicates whether the corresponding phase is sourcing or sinking current. Finally, the switching of the IGBTs are controlled by SW1-SW8. The combinational logic for phase-current selection and the switching of the IGBTs are implemented in programmable gate-array logic chips (GALs) and the source code for programming these GALs are included in Appendix B.1 and B.2 respectively.

5.4.2 Gate Drive Circuits

The half-bridge Intellimod IGBT modules contain integral gate-drive and protection circuitry. A NMV1215DA 12 V to 15 V isolated dc-dc converter is used to provide power
to module’s internal circuitry as shown in Figure 5-13. Gate-drive signals from the microcontroller are supplied to the module via an HP4504 high-speed optocoupler. IGBT fault signals are retrieved through the low-speed PC-819 optocouplers. In principle, only the power supplies to the high-side IGBTs must be isolated, but in the interest of noise minimization and protection of the control board in the event of an IGBT fault, each IGBT is provided with its own isolated supply.

5.4.3 Analog Instrumentation Circuits

The battery voltage, boost converter output voltage, boost inductor current, and two of the motor phase currents are needed for control and display purposes. These low-level analog signals must be acquired from the noisy power circuit environment and transported to the analog/digital control board for amplification and filtering before being used for control. Figure 5-14(a) and (b) show the analog circuits used to acquire the input and output voltages of the inverter. The input voltage and the boost converter output voltage are attenuated via a Zener-protected, high-impedance resistive divider. These signals are transported as differential pairs to the analog/digital control board via a ribbon cable where they are filtered and amplified by Burr Brown INA105 instrumentation amplifiers. The instrumentation amplifiers are preceded by voltage followers for improved common-mode rejection. Both signals are fed into the analog-to-digital (A/D) converter where they are digitized and used for control. The boost converter output voltage signal is also used in conjunction with an analog comparator, as shown in Figure 5-14(c), to generate an output-overvoltage signal which is used to quickly disable the boost converter when its output voltage gets too high.

The boost inductor current is needed for current mode control of the boost converter and also to for controlled charging of the battery during regenerative braking. The current is sensed with a closed-loop Hall-effect current sensor, the output current of which represents the inductor current divided by 2000. The output current of the sensor is converted to a voltage by loading it with a 50 Ω, 1 % precision resistor. The resulting voltage signal,

---

8NMV1215DA is a product of International Power Sources Inc., 200 Butterfield Dr., Ashland, MA 01721
9Connections through the ribbon cable are indicated by dashed lines in the figures
Figure 5-14: Bus voltage and Boost output voltage sensing circuit.

along with its ground, is transported through the ribbon cable to the analog/digital control board where it is filtered and amplified by a Burr Brown INA111 instrumentation amplifier, as shown in Figure 5-15(a). The sensed inductor current is scaled and level shifted for digitization by the A/D converter as shown in Figure 5-15(b). The current signal is also
used to generate an over-current signal which is used as additional protection for the semiconductor switches.

The three motor phase currents always sum to zero. We therefore sense two of the phase currents and generate the third electronically, as shown in Figure 5-16(a). Under normal operation, except during commutation, only two phases conduct current at a time; one being the source and the other being the sink. To regulate the motor current, we close a feedback loop around the current in one of the current-carrying phases. To speed up the commutation of current between phases, we close the feedback loop around the newly commutated motor phase. This control strategy has the effect of raising the voltage available
Figure 5-16: Phase current sensing and selection circuitry.
to build up current in the newly commutated motor phase. The selected phase current in Figure 5-16(b) is scaled and level-shifted for digitization by the A/D converter as shown in Figure 5-16(b). The phase current signal is also used to generate a motor over-current signal to prevent demagnetization of the rotor permanent magnets.

5.4.4 Microcontroller and Peripheral Digital Circuits

The microcontroller used in our experimental prototype is the Motorola MC68HC16Z1, a sixteen-bit microcontroller with 1K of on board Random Access Memory (RAM). A system block diagram showing the connectivity and data flow between the microcontroller to its peripherals is shown in Figure 5-17. The MC68HC16Z1 is equipped with two
programmable pulse-with modulator (PWM) ports, one of which is used to implement duty-cycle control of the buck portion of the inverter, while the other is used to implement current-mode control of the boost section of the inverter.

The MC68HC16Z1 features an eight-channel, 10-bit A/D converter with 10 μs conversion time. This A/D converter is used to sequentially digitize the battery voltage, boost converter’s output voltage, the selected motor phase current, the boost inductor current, the torque and brake user commands which are issued via potentiometers, and the motor’s winding temperatures supplied via thermocouples.

The Queued Serial Peripheral Interphase (QSPI) is used transmit the computed value of peak current for the boost inductor for each control interval to a 12-bit serial D/A converter (DAC). The DAC used is the MAX537 by Maxim Integrated Products. The output of this 12-bit serial DAC is unipolar. Therefore additional external circuitry is needed to implement a bipolar output. As shown in Figure 5-18, the output voltage of the DAC is level shifted and amplified so that the 0 V - 2.5 V output swing of the DAC is mapped to -10 V and +10 V. This voltage is fed into one input of a high-speed comparator with hysteresis which is compared with the sensed boost inductor current to generate a current-mode PWM signal.

User torque commands are input to the controller via a potentiometer on the analog/digital control board. In addition to a torque command input, the user can set the direction of rotation of the motor, engage and disengage the drive via an array of pushbutton switches shown in Figure 5-19.

The MC68HC16Z1 processor comes on an evaluation board (MC68HC16Z1EVB)
which facilitates the downloading of assembled code from the host computer into a 32Kx16 block of RAM (used as pseudo-ROM) for execution by the microcontroller. The evaluation board also facilitates debugging by allowing the user the capability of displaying and modifications of memory locations, stack manipulations, and control over program execution. Figure 5-20 shows a simplified flow chart of the microcode for the experimental prototype described in this thesis. The complete assembly language code is included in Appendix E.
Figure 5-20: Simplified flow chart of microprocessor control code.
Chapter 6

Prototype Fabrication and Experimental Results

Figure 6-1: Schematic diagram of experimental setup.

Figure 6-1 shows a schematic diagram of the experimental system used to evaluate the performance of the prototype wheel motor and inverter. Due to the unavailability of a traction battery at the test site, the inverter was powered off a rectified 3-phase ac source. As shown in the figure, a variable autotransformer (Powerstat\(^1\)) was powered off a 480 V line. The output of the Powerstat, which was variable from 0 to 240 V, was fed into a 30 kVA 3-phase isolation transformer. The output of the isolation transformer was then full-wave rectified by a 3-phase diode bridge and used as the input power source to the inverter. The wheel motor was connected to a dynamometer which was controlled by a four-quadrant

\(^1\)Powerstat is a registered trade-mark of Superior Electric Company.
Figure 6.2: Photographs of experimental setup.

(a) Photograph showing coupling of wheel motor to dynamometer

(b) Photograph of dynamometer showing torque transducer

(c) Photograph of torque meter, inverter, controller board, host computer and other instrumentation equipment

(d) Photograph of inverter for dynamometer
inverter. The torque output, as measured with a strain guage on the dynamometer, was fed to a digital readout. The inverter input voltage, input current, and the motor phase-to-neutral input voltage and phase current were measured with a four-channel Tektronix TDS520 digital oscilloscope\(^2\). The inverter input current and motor phase currents were sensed with a 100 A Tektronix dc current probe. Figure 6-2 shows photographs of the experimental setup.

6.1 Inverter Construction

Figure 6-3 shows the inverter at various stages of construction. To remove switching and conduction losses, the IGBT modules were mounted on a water-cooled aluminum heatsink as shown in Figure 6-3(d). Low-inductance polypropylene snubber capacitors were mounted directly across the terminals of the IGBT modules, as shown in Figure 6-3(a) to minimize voltage overshoot and undershoot from the switching of the power devices. The boost converter output capacitors, shown in Figure 6-3(b), are connected to the IGBTs via the use of L-shaped copper tabs, machined to yield a low inductance connection. The high-voltage buss bar at the output of the boost converter was separated from neighboring conductors by at least 1 cm to prevent flashover from dielectric breakdown. As mentioned in Section 5.2.2, due to the unavailability of appropriate 1200 V capacitors at the time of inverter construction, 1500 V capacitors were used, thus making the boost converter output capacitors 1.5 times larger than necessary.

The isolated dc-dc converters and optocoupler circuits for the gate drives to the IGBTs were wire-wrapped on a perforated board and mounted vertically in close proximity to the IGBTs as shown in Figure 6-3(c). A multiconductor ribbon cable connected this board to the analog/digital control board which in turn connected to the MC68HC16Z1 evaluation board. Detailed schematic diagrams of the inverter's low-voltage electronics are attached in Appendix A.

\(^2\)The phase-to-neutral voltage was measured with a Tektronix P5200 high-voltage differential probe.
Figure 6-3: Prototype inverter at various stages of construction.

(a) Top view showing snubber capacitors and low-inductance connector tabs

(b) Top view showing assembly of boost converter output capacitors

(c) Front view showing gate drives

(d) Side view showing IGBT module, heatsink, and connecting cables
6.2 Motor Construction

The fabrication of the prototype wheel motor was subcontracted to the Superior Electric Company, a motor manufacturing company in Bristol, CT. Figures 6-4 and 6-5 show the motor at various stages of construction. The rotor magnets were machined to the specifications in Figure 3-16 out of Hicorex ND Grade H-97CB material and were supplied by Hitachi Magnetics Corporation in Edmore, MI. The magnets were chromate plated to prevent rusting and had an energy product of 33.44 MGOe. Due to equipment limitations in machining the magnet arcs, each magnet pole was made out of four pieces and glued together. The magnets were then glued on the rotor back-iron as shown in Figure 6-4(a). The rotor back-iron was made out 1018 HF steel supplied by Normag Inc. at Bridgeport, CT. Unlike the stator steel which experiences substantial changes in its magnetic flux density, the rotor turns at the synchronous speed and therefore does not experience substantial variations in magnetic flux density. Therefore the rotor back-iron was made from solid steel, instead of glued laminations, to insure structural integrity. Figure 6-4(b) shows the assembled rotor with the end-bell attached. Due to our desire to keep one end of the motor open for ease of instrumentation, the rotor was cantilevered with bearings on only one side of the stator.

Figure 6-4(c) shows the jig used to skew the stator laminations. The stator laminations were laser-cut from M-15, 29 gauge (14 mils), fully-processed, nonoriented electrical steel. The lamination stack was skewed by one slot pitch. The skewing was accomplished with the aid of two bars, slanted at the skewing angle of 17° from vertical, which guided the laminations as they were stacked. Figure 6-4(d) shows the the glued stator lamination stack mounted for winding. This photograph was taken during the second attempt to wind the motor with a 5-mil Kapton slot-liner. Both attempts resulted in high-potential failures due to abrasion caused by the uneven edges of the skewed lamination stack within slots. To guard against punchthrough of the slot-liner by the edges of the skewed laminations, 10-mil Nomex slot-liners were used. Initial attempts were made to wind the motor with AWG 10 solid square wire, but the wire was difficult to bend and the endturns became unmanageable, as they built up rapidly radially. In order to prevent the radial buildup where the phases cross over each other at the endturns, parallel strands of smaller gauge wires were used so
Figure 6-4: Prototype motor at various stages of construction.
(a) Winding setup showing wire feeding system.  

(b) Schematic showing U-shaped slot-liner and location of thermocouples within the slots.

(c) Wound stator prior to trimming slot-liners and varnishing.  

(d) Assembled motor.

Figure 6-5: Prototype motor at various stages of construction.
that the wires can be flattened at locations where the phases overlap. The price paid for this setup is longer endturns.

The final winding of the prototype motor was done by the author with initial instruction from Dr. Peter Senak, an employee at the Superior Electric Company. The armature conductors were made out of 24 strands of AWG 25 round wire, resulting in an equivalent conductor size of slightly less than AWG 11. The motor was wound in layers of 3 conductors at a time, which were finally connected in series. To minimize the chances of a high-potential failure from winding to winding, U-shaped 10-mil Nomex slot-liners were used to provide additional dielectric isolation between conductors as shown in Figure 6-5(b). Due to several factors including the reduction in slot area by the slot-liners, our desire to keep the axial length of the endturns in check, and lack of experience on the part of the author in winding motors, we were only able to fit 20 out of the designed 21 turns. Thus in winding the 7th layer, one of the three conductors was dropped as indicated by the unshaded conductor section in Figure 6-5(b). The measured winding-to-frame capacitance was 0.01 \( \mu \text{F} \). The wound stator, shown in Figure 6-5(c), was high-potential tested at 2500 V dc from winding to frame, 1500 V dc between phases, and 600 V dc between windings in the same slot. The winding-to-winding high-potential test was performed before the windings were series-connected.

To measure winding temperature, thermocouples were embedded between the 3rd and 4th layer and on top of the 7th layer as shown in Figure 6-5(b). The stator was not vacuum impregnated with a STYCAST-2850FT epoxy encapsulant as was anticipated in Chapter 3 due to unavailability of an appropriate impregnation tank. Furthermore, we wanted to retain the flexibility of re-winding the motor if necessary. Thus, a Sterling 79WS-028 high-bond polyester varnish was used instead. The high-potential tests described above were successfully repeated after the stator was varnished and the rotor and stator were assembled together as shown in Figure 6-5(d).

The cantilever design of the rotor made it difficult to keep a uniform airgap all around the periphery of the stator. Part of the uneveness of the airgap can also be ascribed to machining and manufacturing tolerances. The average measured airgap of the motor was about 15 mils instead of 25 mils. This reduction of the airgap constitutes a 7% increase in the no-load
airgap flux density (given a Carter coefficient of 5%) which more than compensates for the 4.7% reduction in the back-emf constant arising from the dropping of 1 out of 21 turns during the winding of the motor.

The phase-to-phase motor resistance $R_m$ and the phase-to-phase motor inductance $L_m$ were measured by measuring the step response of the motor to a step of voltage excitation across two of the windings with the third open-circuited. The motor resistance was determined as the ratio of the magnitude of the voltage excitation to the final value of the motor current, and the inductance was computed from the time constant of the current waveform. The measured resistance and inductance were $0.47 \, \Omega$ and $3.5 \, \text{mH}$ respectively. The measured inductance was much less than that predicted by the QuickField finite-element program in Chapter 3 from the stored magnetic energy. The discrepancy may be due to inadequate nodes used in mesh generation. The low inductance is actually desirable as it shortens the time for current commutation from one phase to the other. The measured resistance is 38% higher than our original design target of $0.34 \, \Omega$. This increase in motor resistance is due to the reduced copper packing factor and longer endturns resulting from our desire to prevent radial buildup of endturns.

6.3 No-Load Back-EMF

To measure the no-load back-emf, the prototype motor was spun at different speeds and the peak value of the phase-to-neutral voltage was measured. The results, plotted in Figure 6-6 shows a linear relationship between the motor back-emf and speed as expected for a PMSM. The no-load back-emf constant, given by the slope in Figure 6-6, is computed to be $0.76 \, \text{V/rpm}$ or $7.25 \, \text{V/rad-s}$. The no-load back-emf constant is 21% higher than the full-load back-emf value quoted in Figure 3-17 in Chapter 3. Part of this can discrepancy is due to teeth saturation at full-load which reduces the teeth permeability. Also, the magnet temperature in Figure 3-17 was $110^\circ\text{C}$, which reduces the remanent flux of the rotor magnets by 10%. Furthermore, as explained earlier, the reduction of the airgap from 25 mils to 15 mils amount to a 7% increase in the no-load airgap flux density which more than compensates for the 4.7% reduction in the back-emf constant arising from the reduced
Figure 6-6: Variation of no-load peak phase-to-neutral voltage with motor speed.

number of turns.

Figure 6-7 shows the no-load phase-to-neutral back-emf waveforms at three different speeds. These waveforms not only confirm linearity between motor speed and back-emf, but show the trapezoidal nature of the back-emf which is expected from a surface-mounted PMSM with square-wave windings.
Figure 6-7: No-load phase-to-neutral back-emf waveforms at three different speeds.
Figure 6-8: Simulink block diagram and simulated step response of the buck control loop.

### 6.4 Control Loop Simulations

With the measured phase-to-phase motor inductance and motor resistance, feedback gains for the buck and boost digital control loops were determined as described in Section 5.3.2. Since the desired vehicle controller bandwidth is 20 Hz, the bandwidth of the torque controller should be a factor of 10 higher in order to separate the dynamics of the two
Figure 6-9: Simulink block diagram and simulated step response of the boost control loop.

systems [42]. Thus, the closed-loop bandwidth of the torque controller should be at least 200 Hz. The current loop is closed around the newly commutated phase so that the controller provides additional voltage to speed up the commutation of motor current from one phase to another\(^3\). Therefore a closed-loop bandwidth greater than 200 Hz is desirable. The feedback gains for the buck and boost control loops were determined with the Matlab script

\(^3\)Phase current commutation can be further sped up by phase-advance
for pole placement attached in Appendix C. Initial feedback gains were determined by placing the poles in the general area demarcated in Figure 5-9 and the resulting gains were fine-tuned via a simulation of the control system in Simulink.

The poles of the buck control loop were placed at 0.5043±1.2609j and 0.9632 in the z-plane, and the resulting feedback gains were $k_i = 6 \text{ A}^{-1}$, $k_z = 900 \text{ A}^{-1}$, and $k_u = 0$, given an additional gain of 5.11 from the A/D conversion where 100 V and 100 A are converted to 511 (1FF HEX). The pole locations were specially chosen such that previous command feedback gain $k_u$ was zero to reduce computation time. Figure 6-8(a) shows the Simulink block diagram of the buck control loop. The simulated response of the loop to a step in torque command is shown in Figure 6-8(b). The bus voltage for this simulation is 300 V and the motor back-emf is 100 V. Bus voltage feed-forward is implemented in this simulation as shown by the initial non-zero duty cycle. The settling time of the loop to a step input, discounting the long tail, is about 3 ms which is well below the maximum desired settling time of 5 ms.

The poles of the boost control loop were placed at 0.3523±0.408j and 0.4847±0.2385j in the z-plane and the resulting feedback gains were $k_v = 0.046 \text{ A/V}$, $k_i = 0.196 \text{ A/A}$, $k_z = 509 \text{ A/A}$, and $k_u = 1 \text{ A/A}$, including the additional gain of 5.11 from the A/D converter. Figure 6-9(a) shows the Simulink block diagram of the boost control loop. The simulated response of the loop to a step in torque command is shown in Figure 6-9(b). The initial delay of the evolution of the motor current is due to the one-sample delay in the controller, which is also shown in the figure by the average current commanded by the controller. The back-emf for this simulation is 700 V. The settling time of the boost loop to a step input is also about 3 ms which is below the maximum desired settling time of 5 ms.

Figure 6-10 shows an oscillogram of a phase-to-neutral voltage and the corresponding phase current with the buck control loop in operation. This figure corresponds to Experiment #7 in Table 6.1, where the motor produced 250 Nm of torque at 75 rpm. The purpose of this figure is to show that buck control loop operates properly, and that the controller reacts to increase or decrease the phase-to-neutral voltage as necessary to speed up commutation. This oscillogram does not quite indicate the settling time of the control loop since the initial transient includes the commutation period. The phase-to-neutral voltage in the
Figure 6-10: Sample phase-to-neutral voltage and corresponding phase current for buck control loop.

The boost control loop did not operate as envisioned due to the fact that our plant model assumed that the boost inductor always operated in discontinuous conduction. This assumption turned out not to be valid and therefore the plant dynamics changed as the inductor entered into continuous conduction. Sub-harmonic oscillations were also present when the inductor operated in continuous conduction with a duty cycle greater than 0.5. Therefore the controller, as implemented, chattered and its behavior was erratic. Sub-harmonic oscillations in a boost converter can be cured by ramp compensation where a ramp is added to sensed inductor current [59]. The next iteration of the inverter should explore designing the boost converter to run in continuous conduction or using ramp compensation. To operate the boost converter in continuous conduction, additional energy storage is required. The only disadvantage of ramp compensation is that it reduces the dynamic range of the current PWM comparator.

To be able to operate the prototype wheel motor beyond the corner speed, our switching strategy for the power semiconductors was modified to switch the boost converter open
loop with bang-bang duty cycle control to raise its output voltage 100 V above the motor back-emf or 1100 V, whichever is lower. The 3-phase buck converter should only be used for commutation during boost operation. However, because the boost converter was run open-loop, the 3-phase buck converter was used for current control instead. The penalty paid for this control scheme is a maximum additional switching losses in the 3-phase buck converter of about 500 W at the full power of 15 kW, which corresponds to a 3.3% drop in the efficiency of the inverter.

### 6.5 Characterization of Motor-Inverter Efficiency

To obtain an accurate measurement of the efficiency of the wheel motor and inverter, one must include the losses in the bearings of the wheel motor-dynamometer assembly, as the wheel motor must supply those losses. To measure the bearing losses, the wheel motor was spun with its windings open circuited at different speeds, and the torque output of the dynamometer was measured. Figure 6-11 shows the plot of the torque necessary to spin the rotor of the wheel-motor-dynamometer assembly at no load. The quantization of the dynamometer torque meter is 2 Nm. The measured data is fitted to a straight line via least-squares error regression. The no-load torque $T_0$ in Nm is therefore related to the motor RPM by

$$T_0 = 5.5148 + 0.0106 \times \text{RPM} \quad (6.1)$$

Thus $T_0$ must be added to the torque output as read on the dynamometer digital readout in order to obtain the true torque supplied by the wheel motor when spinning the shaft of the dynamometer.

To measure the torque capability of the motor, and the efficiency of the wheel motor and the inverter, the dynamometer was operated at different speeds and the wheel motor was driven to oppose the dynamometer. The speed controller of the dynamometer maintained the shaft speed of the motor at the set rpm, and the torque necessary to keep the shaft speed constant was read from the torque meter. For each experiment the phase-locked signals from the Hall sensors were phase-advanced as necessary to keep the phase current in phase.
Figure 6-11: No-load torques needed to spin the dynamometer-motor assembly at various RPMs.

with the motor back-emf. The temperature of the winding, measured by thermocouples located between the 3rd and 4th layer and on top of the 7th layer are noted. The inverter input bus voltage, input bus current, motor phase-to-neutral voltage and phase current were captured with the oscilloscope as shown in Figure 6-1. The data was post-processed with Matlab. Note that with this setup the inverter bus capacitors not only filtered the ripple current from the switching of the IGBTs but also served as filter capacitors for the 180 Hz input voltage ripple. Thus, current was drawn by the inverter in short bursts at a frequency of 180 Hz, as shown by the sample voltage and current measurement in Figure 6-12.

The input power into the inverter was computed from the bus voltage and bus current data by point-by-point multiplication of the two waveforms and averaging. The output power of the inverter, which is the input power into the motor, was calculated likewise by multiplying the phase-to-neutral voltage by the phase current. The result is averaged over an integral number of cycles and multiplied by 3 to obtain the total power supplied to the three phases of the motor. The no-load back-emf waveforms shown in Figure 6-7 indicate the three phases of the motor are balanced and hence multiplication of the single phase power by 3 is justified. However, from Figure 6-10, we see that the oscilloscope severely
undersamples the 16.44 kHz pulse-width-modulated phase-to-neutral waveform, therefore any calculations based on this waveform are subject to unpredictable errors.

It was anticipated that the fringing fields from the sides of the rotor magnets would be enough to activate the Hall sensors, therefore the rotor magnets did not overhang the stack. Although this was true at no-load, the Hall sensors did not operate reliably when the motor was loaded, due to interference from the armature reaction flux. Thus, it was necessary to constantly fiddle with the orientation of the Hall sensors with respect to the rotor magnets in order to get the controller to operate properly. Due to this problem with the Hall sensors, the GALs on the controller board were re-programmed to shut down all gate-drives when a bad Hall-effect signal set (a combination of all ones or all zeros) was detected. Although this temporary fix allowed data to be taken without inverter or motor malfunction, it is imperative that the magnets overhang the stack in future designs for reliable operation. The operation of the Hall sensors deteriorated severely when the winding temperature exceeded 90°C, hence our inability to obtain experimental data above 800 rpm.

The measured torque, speed, winding temperatures, inverter input bus voltage $V_{bus}$, inverter input power $P_{in}$ and the drive efficiency $\eta_d$, which is the combined efficiency of the motor and inverter are given in Table 6.1\footnote{Due to human operating error, the boost converter switch blew up in the course of our experiments. Thus data for motor operation at 400 rpm and above was taken with the boost 3-phase buck converter operating}
<table>
<thead>
<tr>
<th>Expt. #</th>
<th>Torque (Nm)</th>
<th>Speed (RPM)</th>
<th>Temperature (deg. C)</th>
<th>$V_{bus}$ (Volts)</th>
<th>$P_{in}$ (kW)</th>
<th>$\eta$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$A_7$</td>
<td>$A_3$</td>
<td>$B_7$</td>
<td>$B_3$</td>
</tr>
<tr>
<td>1</td>
<td>157</td>
<td>149</td>
<td>37</td>
<td>38</td>
<td>38</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>159</td>
<td>295</td>
<td>36</td>
<td>37</td>
<td>37</td>
<td>37</td>
</tr>
<tr>
<td>3</td>
<td>28</td>
<td>74</td>
<td>35</td>
<td>35</td>
<td>35</td>
<td>36</td>
</tr>
<tr>
<td>4</td>
<td>159</td>
<td>151</td>
<td>42</td>
<td>42</td>
<td>40</td>
<td>41</td>
</tr>
<tr>
<td>5</td>
<td>158</td>
<td>76</td>
<td>39</td>
<td>40</td>
<td>39</td>
<td>42</td>
</tr>
<tr>
<td>6</td>
<td>388</td>
<td>75</td>
<td>51</td>
<td>52</td>
<td>48</td>
<td>52</td>
</tr>
<tr>
<td>7</td>
<td>250</td>
<td>75</td>
<td>46</td>
<td>48</td>
<td>47</td>
<td>51</td>
</tr>
<tr>
<td>8</td>
<td>199</td>
<td>151</td>
<td>49</td>
<td>47</td>
<td>47</td>
<td>52</td>
</tr>
<tr>
<td>9</td>
<td>148</td>
<td>236</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
</tr>
<tr>
<td>10</td>
<td>234</td>
<td>248</td>
<td>49</td>
<td>50</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>11</td>
<td>30</td>
<td>250</td>
<td>48</td>
<td>48</td>
<td>47</td>
<td>49</td>
</tr>
<tr>
<td>12</td>
<td>33</td>
<td>151</td>
<td>47</td>
<td>47</td>
<td>46</td>
<td>47</td>
</tr>
<tr>
<td>13</td>
<td>153</td>
<td>301</td>
<td>57</td>
<td>59</td>
<td>57</td>
<td>58</td>
</tr>
<tr>
<td>14</td>
<td>75</td>
<td>301</td>
<td>57</td>
<td>56</td>
<td>56</td>
<td>58</td>
</tr>
<tr>
<td>15</td>
<td>29</td>
<td>301</td>
<td>57</td>
<td>56</td>
<td>56</td>
<td>58</td>
</tr>
<tr>
<td>16</td>
<td>253</td>
<td>301</td>
<td>64</td>
<td>66</td>
<td>62</td>
<td>66</td>
</tr>
<tr>
<td>17</td>
<td>309</td>
<td>301</td>
<td>59</td>
<td>61</td>
<td>57</td>
<td>60</td>
</tr>
<tr>
<td>18</td>
<td>412</td>
<td>73</td>
<td>62</td>
<td>63</td>
<td>61</td>
<td>63</td>
</tr>
<tr>
<td>19</td>
<td>429</td>
<td>150</td>
<td>63</td>
<td>65</td>
<td>61</td>
<td>64</td>
</tr>
<tr>
<td>20</td>
<td>402</td>
<td>251</td>
<td>64</td>
<td>66</td>
<td>62</td>
<td>66</td>
</tr>
<tr>
<td>21</td>
<td>93</td>
<td>99</td>
<td>35</td>
<td>34</td>
<td>33</td>
<td>36</td>
</tr>
<tr>
<td>22</td>
<td>325</td>
<td>325</td>
<td>35</td>
<td>36</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>23</td>
<td>94</td>
<td>199</td>
<td>38</td>
<td>38</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>24</td>
<td>274</td>
<td>199</td>
<td>43</td>
<td>41</td>
<td>39</td>
<td>41</td>
</tr>
<tr>
<td>25</td>
<td>366</td>
<td>199</td>
<td>42</td>
<td>43</td>
<td>42</td>
<td>43</td>
</tr>
<tr>
<td>26</td>
<td>100</td>
<td>201</td>
<td>48</td>
<td>48</td>
<td>47</td>
<td>48</td>
</tr>
<tr>
<td>27</td>
<td>46</td>
<td>403</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
</tr>
<tr>
<td>28</td>
<td>128</td>
<td>402</td>
<td>56</td>
<td>57</td>
<td>56</td>
<td>57</td>
</tr>
<tr>
<td>29</td>
<td>238</td>
<td>403</td>
<td>64</td>
<td>66</td>
<td>63</td>
<td>66</td>
</tr>
<tr>
<td>30</td>
<td>81</td>
<td>495</td>
<td>69</td>
<td>71</td>
<td>69</td>
<td>71</td>
</tr>
<tr>
<td>31</td>
<td>169</td>
<td>500</td>
<td>75</td>
<td>78</td>
<td>76</td>
<td>78</td>
</tr>
<tr>
<td>32</td>
<td>49</td>
<td>595</td>
<td>79</td>
<td>80</td>
<td>78</td>
<td>79</td>
</tr>
<tr>
<td>33</td>
<td>50</td>
<td>598</td>
<td>79</td>
<td>81</td>
<td>79</td>
<td>81</td>
</tr>
<tr>
<td>34</td>
<td>148</td>
<td>586</td>
<td>91</td>
<td>91</td>
<td>86</td>
<td>83</td>
</tr>
<tr>
<td>35</td>
<td>89</td>
<td>696</td>
<td>87</td>
<td>90</td>
<td>88</td>
<td>90</td>
</tr>
<tr>
<td>36</td>
<td>83</td>
<td>796</td>
<td>93</td>
<td>93</td>
<td>91</td>
<td>93</td>
</tr>
<tr>
<td>37</td>
<td>106</td>
<td>800</td>
<td>92</td>
<td>95</td>
<td>93</td>
<td>95</td>
</tr>
<tr>
<td>38</td>
<td>106</td>
<td>798</td>
<td>97</td>
<td>100</td>
<td>98</td>
<td>100</td>
</tr>
<tr>
<td>39</td>
<td>127</td>
<td>704</td>
<td>102</td>
<td>106</td>
<td>103</td>
<td>106</td>
</tr>
</tbody>
</table>

Table 6.1: Measured experimental data.
enough to accurately capture the PWM phase-to-neutral voltage, the motor efficiency was computed from the mechanical power output $P_{\text{mech}}$, the Ohmic power loss $P_{\text{Ohm}}$, and an estimated core loss $P_{\text{core}}$. With these parameters, the motor efficiency was estimated as

$$\eta_m = \frac{P_{\text{mech}}}{P_{\text{mech}} + P_{\text{Ohm}} + P_{\text{core}}}$$  \hspace{1cm} (6.2)

where

$$P_{\text{mech}} = \tau \times \frac{2\pi}{60} \text{RPM}$$  \hspace{1cm} (6.3)

and the estimated core loss is given by

$$P_{\text{core}} = 1.30 \times 10^{-5}(1.44 + 1.32 \times 10^{-3}\tau)^2 \text{RPM}^2$$
$$+ 2.59 \times 10^{-2}(1.44 + 1.32 \times 10^{-3}\tau)^{1.79} \text{RPM}$$
$$+ 4.4 \times 10^{-5}\text{RPM}^2 + 7.46 \times 10^{-2}\text{RPM}$$  \hspace{1cm} (6.4)

where RPM indicates the motor speed in rpm. Once the motor efficiency $\eta_m$ was computed, the inverter efficiency $\eta_{inv}$ was calculated from the drive efficiency by the equation

$$\eta_{inv} = 100\frac{\eta_d}{\eta_m}.$$  \hspace{1cm} (6.5)

Note that $P_{\text{Ohm}}$, $P_{\text{core}}$, $\eta_m$ and $\eta_{inv}$ are given in Table 6.2. For completeness, motor losses $P_{l,m}$ computed from the equation

$$P_{l,m} = 3v_{pn}i_p - P_{\text{mech}}$$  \hspace{1cm} (6.6)

where $v_{pn}$ is the motor phase-to-neutral voltage and $i_p$ is the corresponding phase current are also given in Table 6.2. In general, the motor losses computed from Equation (6.6) are higher than the sum of the $I^2R$ losses and the estimated core losses that are also given in Table 6.2. Thus the undersampling of the motor PWM input voltage by the oscilloscope from a higher dc bus voltage as noted in Table 6.1; i.e. the bus was raised by the autotransformer rather than by the boost converter. The bus current measurement in Experiments marked with * in Tables 6.1 and 6.2 are believed to be erroneous as explained in the text. Experiments marked with † are similar others that are also noted in the text.
lead to an overestimation of the motor input power.

The rotary dial of the Tektronix current probe amplifier used to measure the inverter input current sometimes did not make good contact at the 5A/div setting. This is evident from the low bus currents recorded in Experiments 27, 33, 34, 35, and 36. Thus, bus current data for these experiments are erroneous. Since time did not permit these experiments to be repeated, an attempt was made to correct the bus currents in Experiments 33, 34, and 35. Experiment 32 was carried out at practically the same torque-speed point as Experiment 33, except that the bus current setting in Experiment 32 was 10 A/div and that of Experiment 33 was 5A/div. Therefore the bus current measured in Experiment 33 was multiplied by a scale factor of 1.884 to make the drive efficiencies in these two experiments match. Since the bus current setting was not changed for Experiments 33, 34, and 35, this same scale factor was used to correct the measured bus currents in Experiments 34 and 35. Motor and inverter parameters computed with the corrected bus current are preceded by a / in Table 6.1.

Figures 6-13 to 6-15 show plots of the measured drive efficiency, and the computed motor and inverter efficiencies respectively underneath the torque-speed envelope. Note that the winding temperature was not constant for all the experiments, as noted in Table 6.1. Data from Experiments 27 and 36 are not plotted in Figures 6-13 and 6-15 because the bus current measurement errors in these experiments could not be corrected. These experiments are however included in the computed motor efficiency in Figure 6-14 since the bus current measurements were not used in the motor efficiency computations. To avoid clutter, data from Experiments 1, 2, 26, and 37 are not plotted in any of the graphs since they are similar to Experiments 4, 13, 23, and 38 respectively. Overlaying the grayscale density plot of efficiency weights underneath the torque-speed curve over both the urban and highway drive cycles in Figure 2-2 on the plot of motor-inverter efficiency in Figure 6-13, the drive efficiencies from the dc power source to wheel are estimated at 85% and 92% over the urban and highway drive cycles respectively.

Plots of the inverter bus voltage, inverter input power, average motor winding temperature, Ohmic power loss in the armature windings and the estimated core loss in the stator teeth and back-iron at the various points underneath the torque-speed envelope are given in
<table>
<thead>
<tr>
<th>Expt. #</th>
<th>$P_{Ohm}$ (W)</th>
<th>$P_{core}$ (W)</th>
<th>$P_{Ohm} + P_{core}$ (W)</th>
<th>$P_{t,m}$ (W)</th>
<th>$\eta_m$ (%)</th>
<th>$\eta_{inv}$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1†</td>
<td>223</td>
<td>22</td>
<td>246</td>
<td>270</td>
<td>91</td>
<td>87</td>
</tr>
<tr>
<td>2†</td>
<td>251</td>
<td>48</td>
<td>299</td>
<td>499</td>
<td>94</td>
<td>90</td>
</tr>
<tr>
<td>3</td>
<td>8.4</td>
<td>9.8</td>
<td>18</td>
<td>34</td>
<td>92</td>
<td>77</td>
</tr>
<tr>
<td>4</td>
<td>246</td>
<td>23</td>
<td>269</td>
<td>430</td>
<td>90</td>
<td>91</td>
</tr>
<tr>
<td>5</td>
<td>247</td>
<td>11</td>
<td>258</td>
<td>387</td>
<td>77</td>
<td>83</td>
</tr>
<tr>
<td>6</td>
<td>1049</td>
<td>13</td>
<td>1062</td>
<td>767</td>
<td>74</td>
<td>1062</td>
</tr>
<tr>
<td>7</td>
<td>625</td>
<td>12</td>
<td>636</td>
<td>816</td>
<td>75</td>
<td>89</td>
</tr>
<tr>
<td>8</td>
<td>414</td>
<td>23</td>
<td>438</td>
<td>725</td>
<td>88</td>
<td>91</td>
</tr>
<tr>
<td>9</td>
<td>217</td>
<td>37</td>
<td>254</td>
<td>527</td>
<td>94</td>
<td>88</td>
</tr>
<tr>
<td>10</td>
<td>578</td>
<td>41</td>
<td>619</td>
<td>1179</td>
<td>91</td>
<td>96</td>
</tr>
<tr>
<td>11</td>
<td>14</td>
<td>36</td>
<td>50</td>
<td>258</td>
<td>94</td>
<td>75</td>
</tr>
<tr>
<td>12</td>
<td>15</td>
<td>21</td>
<td>36</td>
<td>143</td>
<td>93</td>
<td>75</td>
</tr>
<tr>
<td>13</td>
<td>247</td>
<td>48</td>
<td>296</td>
<td>706</td>
<td>94</td>
<td>91</td>
</tr>
<tr>
<td>14</td>
<td>71</td>
<td>46</td>
<td>117</td>
<td>537</td>
<td>95</td>
<td>87</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>45</td>
<td>60</td>
<td>413</td>
<td>94</td>
<td>70</td>
</tr>
<tr>
<td>16</td>
<td>679</td>
<td>52</td>
<td>731</td>
<td>1104</td>
<td>92</td>
<td>95</td>
</tr>
<tr>
<td>17</td>
<td>1049</td>
<td>54</td>
<td>1103</td>
<td>1699</td>
<td>90</td>
<td>93</td>
</tr>
<tr>
<td>18</td>
<td>1794</td>
<td>12</td>
<td>1806</td>
<td>2116</td>
<td>63</td>
<td>88</td>
</tr>
<tr>
<td>19</td>
<td>2121</td>
<td>27</td>
<td>2148</td>
<td>2801</td>
<td>76</td>
<td>91</td>
</tr>
<tr>
<td>20</td>
<td>1932</td>
<td>47</td>
<td>1978</td>
<td>2966</td>
<td>84</td>
<td>95</td>
</tr>
<tr>
<td>21</td>
<td>83</td>
<td>14</td>
<td>97</td>
<td>168</td>
<td>91</td>
<td>98</td>
</tr>
<tr>
<td>22</td>
<td>1036</td>
<td>16</td>
<td>1052</td>
<td>1369</td>
<td>76</td>
<td>91</td>
</tr>
<tr>
<td>23</td>
<td>91</td>
<td>29</td>
<td>121</td>
<td>348</td>
<td>94</td>
<td>88</td>
</tr>
<tr>
<td>24</td>
<td>731</td>
<td>33</td>
<td>764</td>
<td>1084</td>
<td>88</td>
<td>93</td>
</tr>
<tr>
<td>25</td>
<td>1414</td>
<td>35</td>
<td>1449</td>
<td>2232</td>
<td>84</td>
<td>92</td>
</tr>
<tr>
<td>26†</td>
<td>103</td>
<td>30</td>
<td>133</td>
<td>278</td>
<td>94</td>
<td>93</td>
</tr>
<tr>
<td>27††</td>
<td>25</td>
<td>64</td>
<td>89</td>
<td>316</td>
<td>96</td>
<td>1012</td>
</tr>
<tr>
<td>28</td>
<td>43</td>
<td>67</td>
<td>110</td>
<td>-2392</td>
<td>98</td>
<td>91</td>
</tr>
<tr>
<td>29</td>
<td>628</td>
<td>72</td>
<td>115</td>
<td>1478</td>
<td>93</td>
<td>95</td>
</tr>
<tr>
<td>30</td>
<td>71</td>
<td>83</td>
<td>154</td>
<td>402</td>
<td>96</td>
<td>94</td>
</tr>
<tr>
<td>31</td>
<td>318</td>
<td>89</td>
<td>408</td>
<td>874</td>
<td>96</td>
<td>97</td>
</tr>
<tr>
<td>32</td>
<td>25</td>
<td>102</td>
<td>127</td>
<td>128</td>
<td>96</td>
<td>96</td>
</tr>
<tr>
<td>33††</td>
<td>28</td>
<td>103</td>
<td>131</td>
<td>243</td>
<td>96</td>
<td>181</td>
</tr>
<tr>
<td>34*</td>
<td>247</td>
<td>107</td>
<td>138</td>
<td>1086</td>
<td>96</td>
<td>176</td>
</tr>
<tr>
<td>35*</td>
<td>91</td>
<td>128</td>
<td>146</td>
<td>3851</td>
<td>97</td>
<td>97</td>
</tr>
<tr>
<td>36*</td>
<td>25</td>
<td>148</td>
<td>173</td>
<td>37</td>
<td>96</td>
<td>101</td>
</tr>
<tr>
<td>37†</td>
<td>144</td>
<td>155</td>
<td>300</td>
<td>1101</td>
<td>97</td>
<td>96</td>
</tr>
<tr>
<td>38</td>
<td>142</td>
<td>155</td>
<td>297</td>
<td>815</td>
<td>97</td>
<td>97</td>
</tr>
<tr>
<td>39</td>
<td>216</td>
<td>134</td>
<td>350</td>
<td>1280</td>
<td>96</td>
<td>94</td>
</tr>
</tbody>
</table>

Table 6.2: Computed performance parameters from experimental data.
6.5.1 Thermal Performance

As described in Chapter 3, the wheel motor is completely sealed in its operating environment. One side of the prototype wheel motor was however left open for ease of experimentation. Furthermore the motor, through its bearings and mechanical couplings, was thermally connected to the dynamometer. Therefore it was impossible to obtain any thermal data representative of motor operation in the hub of a wheel. One point that is worth noting is that the winding temperatures recorded in Table 6.1 indicate that temperature between the 3rd and 4th winding layers, measured by thermocouples A3 and B3, are higher than that on top of the 7th layer, as measured by thermocouples A7 and B7. Thus the hot spot has shifted to the center of the windings. This result is expected since the airgap is no more adiabatic as was assumed in Section 3.5.4.

Although we were unable to obtain actual temperatures representative of motor operation in the hub of a wheel, Figure 6-21 is of special importance since it shows the amount of heat that must be removed at the various operating point. Note from Figure 6-13 that the motor operates on the average at about 1/5 of its constant torque rating over the drive cycles. As Figure 6-21 shows, the thermal burden of the wheel motor is significantly reduced by the ability to lower the motor current at high speeds and therefore the motor can be air cooled while producing substantial horsepower. This reduced thermal burden is one of the benefits of the buck-boost inverter.

6.6 Summary

A 450 Nm, 20 hp, 25.1 kg permanent-magnet wheel motor and a microprocessor-controlled novel power-electronic drive that incorporates a bidirectional boost converter has been built and tested in a laboratory environment. The wheel motor-inverter combination, designed to operate up to 1300 rpm was tested at various points underneath its torque-speed curve up to 800 rpm. Experimental data for operation above 800 rpm were unobtainable due to unreliable operation of the Hall-effect magnetic field sensors at high temperatures. Due to
human operating error, one of the switches of the boost converter was destroyed therefore operation above a bus voltage of 300 V was achieved by raising the bus voltage with an autotransformer. The drive efficiencies, from the dc power source to wheel, are estimated at 85% over the Federal Urban Drive Schedule and 92% over the Federal Highway Drive Schedule.
Figure 6-13: Measured efficiency of the inverter motor combination in percent.
Figure 6-14: Computed efficiency of motor in percent.
Figure 6-15: Computed efficiency of inverter in percent.
Figure 6.16: Inverter input bus voltage in Volts.
Figure 6-18: Average armature winding temperature in deg. C.
Figure 6-19: Ohmic power dissipation in armature windings in Watts.
Figure 6-20: Estimated core loss in stator steel in Watts.
Figure 6-21: Total heat generated in motor in Watts.
Chapter 7

Summary and Conclusions

7.1 Thesis Summary

From an applications viewpoint, this thesis develops a permanent-magnet synchronous motor and a microprocessor-controlled novel power-electronic inverter for use in direct-drive EV propulsion. The development raises a variety of engineering and scientific issues which are addressed as follows.

- Environmental legislation by the CARB to improve air quality in urban areas, coupled with the PNGV initiative by the Federal government and the USCAR, has rekindled interest in the development of EVs as a competitive means of transport to ICEVs. An EV driven by wheel motors has improved powertrain efficiency due to the elimination of losses in the drive line and mechanical transmission; see Figure 1-2. A wheel motor powertrain also offers advantages in the packaging and manufacturing of the vehicle. The elimination of intermediate gearing requires the wheel motors to produce the high torques necessary to accelerate the vehicle in the limited space available in the hub of the wheel. There is also a strong need to reduce the mass of the wheel motors to reduce the unsprung weight of the vehicle so that the ride quality of the vehicle is not compromised. Thus the wheel motor offers a significant technical challenge.

- The use of newly available Nd-Fe-B magnets permits the design of efficient high-torque, low-mass motors and the availability of high-voltage, fast-switching IGBTs
allow these motors and their inverters to operate efficiently at high voltage and reduced currents. The combination can lead to a very low weight and efficient drive, as demonstrated in Chapter 6.

- Unlike an industrial motor, which is designed to operate continuously at rated torque, a wheel motor typically operates at an average of one-fifth of its peak torque rating over a drive cycle. Thus, the ability to produce low torques efficiently at medium and high speeds is key for drive-cycle efficiency maximization.

- The desire to reduce no load losses calls for the exploration of a slotless motor for the wheel motor application. With a slotless armature the electromagnetic airgap is large, since it includes the thickness of the airgap windings. Thus, there is the need to explore magnet arrangements which maximize radial airgap flux density per volume of magnet material used. The torque-production characteristics of a Halbach and conventional rotor magnet arrangement, with and without a permeable rotor back-iron, operating with a magnetically-backed slotless armature was studied in Chapter 2. The results of this study showed that a more economic motor design can be obtained with a slotted armature, and that with a slotted armature the preferred magnet arrangement is the conventional arrangement for the EV application. The study also explained, in an intuitive manner, why and when to use a Halbach array in the design of PMSMs.

- In Chapter 3, we designed a slotted motor with a conventional magnet arrangement that is capable of producing 450 Nm of torque at 110°C. The design methodology was to produce the highest torque per Ohmic power dissipation in the armature windings while keeping magnet cost in check. The motor was optimized in terms of the number of pole-pairs and the thickness of the armature windings. Low-loss M-15, 29 gauge nonoriented electrical steel was used in the stator construction to reduce the no-load losses in the motor. Unlike the stator back-iron which experiences significant changes in magnetic flux density during motor operation, the rotor back-iron was made out of solid 1018 HF steel for enhanced structural integrity. The active mass of the wheel motor is 25.1 kg.
For efficient operation of the power-electronic inverter, the number of turns of the motor was chosen so that the inverter operates at high voltage and reduced current. Efficient 4:1 speed range operation is achieved with a 3-phase inverter with a bidirectional boost converter in its dc link. By using this novel inverter topology we avoid the need for flux weakening which compromises drive efficiency. The cooling requirements of the motor are also reduced. In addition to improved drive-cycle efficiency, there is a factor of 1.5 reduction in the installed V-A capacity over a low-voltage/high-current inverter that does not utilize flux weakening to achieve wide-speed range operation, and the snubbing requirements are also mitigated. These issues are discussed in Chapter 4.

Detailed design of the power-electronic inverter and the microprocessor controller are given in Chapter 5.

The wheel motor and novel inverter were fabricated and tested at the Superior Electric Company in Bristol, CT, as described in Chapter 6. The efficiency of the drive (inverter and motor) over the Urban and Highway drive cycles were estimated at 85% and 92% respectively. The packaging of the motor and its assembly on the dynamometer did not permit a meaningful thermal characterization of the motor.

### 7.2 Thesis Conclusions

There are three major conclusions that can be drawn from this thesis. The first is that efficient high-torque, low-mass wheel motors are possible. A viable design approach for the motor is to maximize the peak torque produced per Ohmic power dissipated in the armature windings. A slotted stator is desirable in reducing magnet mass and a high number of poles is required to reduce endturn length and the thickness of the rotor and stator back-iron. The use of low-loss electrical steel with thin laminations (14 mils in this thesis) help to reduce eddy-current and hysteresis losses in the stator teeth. The rotor back-iron may be made of solid electrical steel.

Secondly, the use of a 3-phase inverter with a bidirectional boost converter in the dc
link permits efficient operation of the high-voltage/low-current power-electronic inverter. By rating the semiconductor switches to withstand the highest back-emf of the motor, catastrophic failures where the motor injects large currents into the dc power source such as a traction battery is avoided. The cooling requirements of the motor are reduced by not using flux-weakening as a means to obtain wide speed range operation. The cost of the new inverter is comparable to, if not less than, that of a standard 3-phase voltage-source inverter.

The third conclusion pertains to when to use a Halbach magnet arrangement in the design of a PMSM. Halbach arrays are superior when one is precluded from using a magnetic rotor backing. In such applications the Halbach magnet arrangement produces higher torque than a conventional magnet arrangement. If a magnetic rotor backing can be used, then one does better by using a conventional magnet arrangement when the rotor magnets are thin. Beyond a certain thickness of rotor magnets (this depends on the geometry of the motor in question) a Halbach magnet arrangement becomes beneficial.

7.3 Recommendations for Future Work

In our experimentation, we discovered that the SS461A Hall sensors could not be made to operate reliably with only the fringing fields from the magnets. Therefore the next iteration of the wheel motor design should have the magnets overhang the stack by a couple of millimeters so that the Hall sensors can be located underneath the overhang. This arrangement will insure reliable operation of the Hall sensors over the temperature range of operation of the motor without false triggering from the armature reaction flux. Another way of determining rotor magnet position with respect to the stator windings is to paint the edges of alternating magnets and use an optical emitter-detector pair. Although this approach is less robust than using the Hall magnetic field sensors due to eventual degradation of the optical surfaces from dirt, it may be necessary to implement such an optical shaft encoder to insure reliable operation of the prototype wheel motor. The third approach, which would be ideal if it can be made to work, is to commutate the motor based on the back-emf of the motor, measured through the non-conducting phase. This approach eliminates the need for an additional sensor within the motor housing.
Our original intent of reducing switching losses in the three half-bridges that drive the motor in the boost mode when the motor back-emf was above the dc bus voltage was unachievable since it was not possible to close a robust control loop around the boost converter. This is because the boost inductor sometimes operated in continuous conduction, thus changing the plant dynamics. The absence of ramp compensation also resulted in sub-harmonic oscillations which caused the inductor to chatter. The next iteration of the inverter design should explore adding a ramp to the sensed boost inductor current or operating the boost inductor in continuous conduction. Although adding a ramp to the boost inductor current eliminates sub-harmonic oscillations, ramp compensation has the undesirable effect of reducing the dynamic range of the current-mode PWM comparator. Operating the boost inductor in continuous conduction will prevent the plant dynamics from changing in various regimes of the boost converter's operation. The cost is additional energy storage in the boost inductor.

Due to the lack of proper packaging and mounting of the prototype wheel motor on the dynamometer, it was impossible to obtain meaningful data that could predict the thermal performance of the motor. These measurements remain to be done.

Future designs of the wheel motor should also explore skewing the magnets instead of the slots to reduce cogging, as the skewed slots expose sharp lamination edges which can cut easily through slot liners. Furthermore, unskewed slots will make the motor easier to wind, require less wire and reduce Ohmic losses in the armature windings.

Regenerative braking of the motor was not implemented. This remains to be done. Last but not least, the MC68HC16Z1 is fast enough for the control of a single wheel motor, and therefore the motor commutation and phase current selection logic which are currently implemented in GALs can be implemented in software within the microcontroller.
Appendix A

Circuit Schematics

This Appendix contains the schematics of the low-voltage circuitry on the gate-drive board, control board attached to the MC68HC16Z1 evaluation board and the user interface keypad. Descriptions of these circuits and how they operate are given in Chapter 5.
Figure A-1: Circuitry on gate-drive board.
Figure A-2: Voltage sensing circuitry.
Figure A-3: Boost inductor current processing circuitry.
Figure A-4: Motor phase current processing circuitry.
Figure A-5: Gate-Array Logic chips and power supplies for analog/digital board.
Figure A-6: Circuitry on user keypad and display board
Figure A-7: Simplified schematic of microcontroller showing pins used
Appendix B

GAL22v10 Programming Files

To simplify the wiring of the analog/digital board for the prototype inverter, most of the digital glue logic was implemented with electrically erasable programmable Gate Array Logic (GAL). A total of four GAL22V10 devices were used in the experimental prototype. This chapter contains the source code describing the logic configuration of the devices. A schematic chip diagram is attached at the end of each source code.

B.1 Phase Current Selection Logic

The function of this GAL is to generate the phase current select signals SELA, SELB, SELC, and phase current negate signal, NEG, that are used to determine the motor phase current around which to close the current feedback loop. The truth table of the logic implemented in this GAL is given in Figure 5-12 in Chapter 5. A fault signal is generated in the event of motor-phase overcurrent, boost-inductor overcurrent, boost-converter overvoltage, or a bad Hall-effect signal combination - which is all zeroes or all ones. The fault signal is used to shut down the IGBTs and is cleared by pushing a button on the user interface keypad.

Name select;
Partno CA0006;
Date 11/28/95;
Revision 01;
Designer ofori;
Company MIT;
Assembly None;
Location None;
Device P22V10;

/*****************************/
/* Phase current select combinational logic generator */
/*****************************/

/*****************************/
/* Allowable Target Device Types : GAL22V10 */
/*****************************/

/** Inputs **/

PIN 1 = INH; /* HIGH TO TURN OFF GATE DRIVES */
PIN 2 = NO_IGBT_FAULT; /* NO IGBT FAULT */
PIN 3 = FORW; /* SPIN MOTOR FORWARD OR REVERSE */
PIN 4 = MOT; /* MOTOR OR REGEN, BUT NOT USED*/
PIN 5 = PL_HEA; /* HALL EFFECT A */
PIN 6 = PL_HEB; /* HALL EFFECT B */
PIN 7 = PL_HEC; /* HALL EFFECT C */
PIN 8 = PWMB; /* BUCK PWM SIGNAL */
PIN 9 = VBOOST_IN_RANGE; /* HIGH WHEN VBOOST IS IN RANGE */
PIN 10 = IPHASE_IN_RANGE; /* HIGH WHEN PHASE CURRENT IS IN RANGE */
PIN 11 = ILB_IN_RANGE; /* HIGH WHEN BOOST IND. CURRENT IS IN RANGE */
PIN 13 = CLEAR_FAULT; /* HIGH TO CLEAR ALL FAULTS */

/** Outputs **/

PIN 23 = SELB; /* SELECT PHASE B CURRENT */
PIN 22 = SELC; /* SELECT PHASE C CURRENT */
PIN 21 = SELA; /* SELECT PHASE A CURRENT */
PIN 20 = NEG_PHASE; /* HIGH WHEN SENSED PHASE CURRENT IS */
/* TO BE NEGATED */
PIN 19 = PWMB_BAR; /* INVERTED PWM TO TRIGGER BUCK RAMP */
PIN 18 = FAULT; /* HIGH DURING IGBT FAULT, BOOST INDUCTOR ... */
/* OVER CURRENT, BOOST OVERVOLTAGE, PHASE ... */
/* OVER CURRENT, OR BAD HALL EFFECT COMBINATION */
PIN 17 = NO_FAULT; /* HIGH WHEN NO FAULT HAS OCCURED */
PIN 16 = BAD_HALL_SIGNAL;
PIN 15 = GOOD_HALL_SIGNAL;

/** Declarions and Intermediate Variable Definitions **/
BAD_HALL_EFFECT = (PL_HEA & PL_HEB & PL_HEC) #
((!PL_HEA & !PL_HEB & !PL_HEC);

/** Logic Equations **/

SELA = ((!PL_HEA & PL_HEB & !PL_HEC) # (PL_HEA & !PL_HEB & PL_HEC);

SELB = ( !PL_HEA & !PL_HEB & PL_HEC) # ( PL_HEA & PL_HEB & !PL_HEC);
SELC = ( !PL_HEA & PL_HEB & PL_HEC) # ( PL_HEA & !PL_HEB & !PL_HEC);  

NEG_PHASE = ( !FORW & !PL_HEA & PL_HEB & PL_HEC) #  
( !FORW & PL_HEA & !PL_HEB & PL_HEC) #  
( !FORW & PL_HEA & PL_HEB & !PL_HEC) #  
( FORW & PL_HEA & !PL_HEB & !PL_HEC) #  
( FORW & !PL_HEA & PL_HEB & !PL_HEC) #  
( FORW & !PL_HEA & !PL_HEB & PL_HEC) ;

PWMB_BAR = !PWMB;

FAULT = !NO_FAULT # !I_PHASE_IN_RANGE #  
!ILB_IN_RANGE # BAD_HALL_EFFECT # !NO_IGBTFAULT;  
NO_FAULT = !FAULT # CLEAR_FAULT;

GOOD_HALL_SIGNAL = !BAD_HALL_SIGNAL # CLEAR_FAULT;

BAD_HALL_SIGNAL = !GOOD_HALL_SIGNAL # BAD_HALL_EFFECT;

Chip Diagram

<table>
<thead>
<tr>
<th>select</th>
</tr>
</thead>
</table>
| INH    | 1  
| NO_IGBT_FAULT | 2  
| FORW   | 3  
| MOT    | 4  
| PL_HEA | 5  
| PL_HEB | 6  
| PL_HEC | 7  
| PWMB   | 8  
| VBOOST_IN_RANGE | 9  
| I_PHASE_IN_RANGE | 10  
| ILB_IN_RANGE | 11  
| GND    | 12  
|        | 13  

| Vcc   | 24  
| SELB  | 23  
| SLEC  | 22  
| SELA  | 21  
| Vcc   | 24  
| PWMB_BAR | 19  
| Vcc   | 24  
| Vcc   | 24  
| Vcc   | 24  
| Vcc   | 24  

184
B.2 IGBT Switching Logic

This GAL determines the switching status of all the intellimod IGBTs based on the truth table in Figure 5-12. The SW1 - SW8 signals generated by this GAL is buffered by an octal line driver (74HCT244) and used to drive the gates of the intellimod IGBTs. The inhibit (INH) signal, when high, turns off all the IGBTs. The phase-locked Hall-effect signals (PL_HEA - PL_HEC) in conjunction with the motor rotation direction signal (MOT) are used to commutate the motor. Two different IGBT switching strategy were implemented; one with current-mode control of the boost converter where the 3-phase buck converter just operates as a commutator and the other where the boost converter was switched open loop with bang-bang control and the 3-phase buck converter was used to regulate the motor current. The code for both versions of the GAL are given below.

B.2.1 Switching Logic for Current-Mode Control of Boost Converter

```plaintext
Name  IGBT_SW1;
Partno CA0006 ;
Date  11/28/95;
Revision 01;
Designer ofori;
Company MIT;
Assembly None;
Location None;
Device P22V10;

/************************************************************************
/* Switch states combinational logic generator */
/* */
/************************************************************************

/************************************************************************
/* Allowable Target Device Types : GAL22V10 */
/************************************************************************

/** Inputs **/

PIN 1 = INH; /* HIGH TO TURN OFF GATE DRIVES */
PIN 2 = NOT_USED; /* CURRENTLY WIRED TO LOCK */
PIN 3 = FORW; /* SPIN MOTOR FORWARD OR REVERSE */
PIN 4 = MOT; /* MOTOR OR REGEN */
PIN 5 = PL_HEA; /* HALL EFFECT A */
PIN 6 = PL_HEB; /* HALL EFFECT B */
```
PIN 7 = PL_HEC; /* HALL EFFECT C */
PIN 8 = BUCK; /* WHEN BATTERY VOLTAGE > BACK EMF */
PIN 9 = PWM_BUCK; /* PWM SIGNAL FOR BUCK CONVERTER */
PIN 10 = PWM_BOOST; /* PWM SIGNAL FOR BOOST CONVERTER */
PIN 11 = VBOOST_OK; /* HIGH WHEN VBOOST IS BELOW MAX VALUE */
PIN 13 = NO_FAULT; /* HIGH WHEN THERE IS NO FAULT */

/* FAULT HERE MEANS BOOST INDUCTOR OVERCURRENT, BOOST OVERVOLTAGE */
/* PHASE OVERCURRENT, BAD HALL EFFECT SIGNAL, AND IGBT FAULT */
/* FAULT IS CLEAR BY PUSHING CLEAR FAULT BUTTON ON KEYPAD */

/** Outputs /**
PIN 23 = SW1; /* TURNON SIGNAL FOR BOOST TOP SWITCH */
PIN 22 = SW2; /* TURNON SIGNAL FOR BOOST BOT SWITCH */
PIN 21 = SW3; /* TURNON SIGNAL FOR PHASE A TOP SWITCH */
PIN 20 = SW4; /* TURNON SIGNAL FOR PHASE A BOT SWITCH */
PIN 19 = SW5; /* TURNON SIGNAL FOR PHASE B TOP SWITCH */
PIN 18 = SW6; /* TURNON SIGNAL FOR PHASE B BOT SWITCH */
PIN 17 = SW7; /* TURNON SIGNAL FOR PHASE C TOP SWITCH */
PIN 16 = SW8; /* TURNON SIGNAL FOR PHASE C BOT SWITCH */
PIN 15 = FAULT; /* HIGH WHEN THERE IS A FAULT */

/** Declarations and Intermediate Variable Definitions /**

DRIVE_OK = !INH & NO_FAULT;

/** Logic Equations /**

SW1 = (DRIVE_OK & !MOT & !PL_HEA & PL_HEC & PWM_BOOST) #
 (DRIVE_OK & !MOT & !PL_HEA & PL_HEB & PWM_BOOST) #
 (DRIVE_OK & !MOT & PL_HEA & !PL_HEC & PWM_BOOST) #
 (DRIVE_OK & !MOT & PL_HEA & !PL_HEB & PWM_BOOST) #
 (DRIVE_OK & BUCK & !MOT & !PL_HEA & PL_HEC) #
 (DRIVE_OK & BUCK & !MOT & !PL_HEA & PL_HEB) #
 (DRIVE_OK & BUCK & !MOT & PL_HEA & !PL_HEC) #
 (DRIVE_OK & BUCK & !MOT & PL_HEA & !PL_HEB);

/* VBOOST_OK DISABLES BOOST CONVERTER WHEN BUS GETS TOO HIGH */
/* CURRENTLY REDUNDANT BECAUSE OF THE DEFINITION OF FAULT */
/* IF ALL WORKS WELL, VBOOST_OK FROM FAULT DEFINITION */

SW2 = (DRIVE_OK & !BUCK & (MOT & VBOOST_OK) & !PL_HEA & PL_HEC & PWM_BOOST) #
 (DRIVE_OK & !BUCK & (MOT & VBOOST_OK) & !PL_HEA & PL_HEB & PWM_BOOST) #
 (DRIVE_OK & !BUCK & (MOT & VBOOST_OK) & PL_HEA & !PL_HEC & PWM_BOOST) #
 (DRIVE_OK & !BUCK & (MOT & VBOOST_OK) & PL_HEA & !PL_HEB & PWM_BOOST);

SW3 = (DRIVE_OK & FORW & MOT & PL_HEA & !PL_HEB & PWM_BUCK) #
 (DRIVE_OK & FORW & MOT & !PL_HEA & PL_HEB & PWM_BUCK) #
 (DRIVE_OK & FORW & !BUCK & MOT & PL_HEA & !PL_HEB) #
 (DRIVE_OK & FORW & !BUCK & MOT & !PL_HEA & PL_HEB);
SW4 = (DRIVE_OK & !FORW & !MOT & !PL_HEA & PL_HEB & PWM_BUCK) #
    (DRIVE_OK & !FORW & !BUCK & !MOT & !PL_HEA & PL_HEB) #
    (DRIVE_OK & FORW & MOT & !PL_HEA & PL_HEB) #
    (DRIVE_OK & FORW & BUCK & !PL_HEA & PL_HEB) #
    (DRIVE_OK & !MOT & PL_HEA & !PL_HEB & PWM_BUCK) #
    (DRIVE_OK & !BUCK & !MOT & PL_HEA & !PL_HEB) #
    (DRIVE_OK & !FORW & PL_HEA & !PL_HEB);

SW5 = (DRIVE_OK & !FORW & MOT & !PL_HEB & PL_HEC & PWM_BUCK) #
    (DRIVE_OK & !FORW & !BUCK & MOT & !PL_HEB & PL_HEC) #
    (DRIVE_OK & FORW & MOT & PL_HEB & !PL_HEC & PWM_BUCK) #
    (DRIVE_OK & FORW & !BUCK & MOT & PL_HEB & !PL_HEC);

SW6 = (DRIVE_OK & !MOT & PL_HEB & !PL_HEC & PWM_BUCK) #
    (DRIVE_OK & !MOT & !PL_HEB & PL_HEC & PWM_BUCK) #
    (DRIVE_OK & FORW & !PL_HEB & PL_HEC) #
    (DRIVE_OK & !BUCK & !MOT & PL_HEB & !PL_HEC) #
    (DRIVE_OK & !BUCK & !MOT & !PL_HEB & PL_HEC) #
    (DRIVE_OK & !FORW & PL_HEB & !PL_HEC);

SW7 = (DRIVE_OK & !FORW & MOT & PL_HEA & !PL_HEC & PWM_BUCK) #
    (DRIVE_OK & !FORW & !BUCK & MOT & PL_HEA & !PL_HEC) #
    (DRIVE_OK & FORW & MOT & !PL_HEA & PL_HEC & PWM_BUCK) #
    (DRIVE_OK & FORW & !BUCK & MOT & !PL_HEA & PL_HEC);

SW8 = (DRIVE_OK & !MOT & !PL_HEA & PL_HEC & PWM_BUCK) #
    (DRIVE_OK & !MOT & PL_HEA & !PL_HEC & PWM_BUCK) #
    (DRIVE_OK & FORW & PL_HEA & !PL_HEC) #
    (DRIVE_OK & !BUCK & !MOT & !PL_HEA & PL_HEC) #
    (DRIVE_OK & !BUCK & !MOT & PL_HEA & !PL_HEC) #
    (DRIVE_OK & !FORW & !PL_HEA & PL_HEC);

FAULT = !NO_FAULT ;
### Chip Diagram

<table>
<thead>
<tr>
<th>IGBT_SW1</th>
</tr>
</thead>
<tbody>
<tr>
<td>INH x--</td>
</tr>
<tr>
<td>NOT_USED x--</td>
</tr>
<tr>
<td>FORW x--</td>
</tr>
<tr>
<td>MOT x--</td>
</tr>
<tr>
<td>PL_HEA x--</td>
</tr>
<tr>
<td>PL.HEB x--</td>
</tr>
<tr>
<td>PL.HEC x--</td>
</tr>
<tr>
<td>BUCK x--</td>
</tr>
<tr>
<td>PWM_BUCK x--</td>
</tr>
<tr>
<td>PWM_BOOST x--</td>
</tr>
<tr>
<td>VBOOST_OK x--</td>
</tr>
<tr>
<td>GND x--</td>
</tr>
</tbody>
</table>

---

#### B.2.2 Switching Logic for Current-Mode Control of Boost Converter

Name  IGBT_SW2;
Partno CA0006;
Date  11/28/95;
Revision 01;
Designer ofori;
Company MIT;
Assembly None;
Location None;
Device P22V10;

/***********************************************************/
/* Switch states combinational logic generator          */
/* for the implementation where the boost converter is switched */
/* open loop                                          */
/***********************************************************/

/***********************************************************/
/* Allowable Target Device Types : GAL22V10            */
/***********************************************************/

/** Inputs **/

PIN 1 = INH; /* HIGH TO TURN OFF GATE DRIVES */
PIN 2 = NOT_USED;  /* CURRENTLY WIRED TO LOCK */
PIN 3 = FORW;  /* SPIN MOTOR FORWARD OR REVERSE */
PIN 4 = MOT; /* MOTOR OR REGEN */
PIN 5 = PL_HEA; /* HALL EFFECT A */
PIN 6 = PL_HEB; /* HALL EFFECT B */
PIN 7 = PL_HEC; /* HALL EFFECT C */
PIN 8 = BUCK; /* WHEN BATTERY VOLTAGE > BACK EMF */
PIN 9 = PWM_BUCK; /* PWM SIGNAL FOR BUCK CONVERTER */
PIN 10 = PWM_BOOST; /* PWM SIGNAL FOR BOOST CONVERTER */
PIN 11 = VBOOST_OK; /* HIGH WHEN VBOOST IS BELOW MAX VALUE */
PIN 13 = NO_FAULT; /* HIGH WHEN THERE IS NO FAULT */

/* FAULT HERE MEANS BOOST INDUCTOR OVERCURRENT, BOOST OVERVOLTAGE */
/* PHASE OVERCURRENT, BAD HALL EFFECT SIGNAL, AND IGBT FAULT */
/* FAULT IS CLEAR BY PUSHING CLEAR FAULT BUTTON ON KEYPAD */

/** Outputs **/
PIN 23 = SW1; /* TURNON SIGNAL FOR BOOST TOP SWITCH */
PIN 22 = SW2; /* TURNON SIGNAL FOR BOOST BOT SWITCH */
PIN 21 = SW3; /* TURNON SIGNAL FOR PHASE A TOP SWITCH */
PIN 20 = SW4; /* TURNON SIGNAL FOR PHASE A BOT SWITCH */
PIN 19 = SW5; /* TURNON SIGNAL FOR PHASE B TOP SWITCH */
PIN 18 = SW6; /* TURNON SIGNAL FOR PHASE B BOT SWITCH */
PIN 17 = SW7; /* TURNON SIGNAL FOR PHASE C TOP SWITCH */
PIN 16 = SW8; /* TURNON SIGNAL FOR PHASE C BOT SWITCH */
PIN 15 = FAULT; /* HIGH WHEN THERE IS A FAULT */

/** Declarations and Intermediate Variable Definitions **/

DRIVE_OK = !INH & NO_FAULT & MOT & BUCK;

/** Logic Equations **/

SW1 = 'b'0;

/* VBOOST_OK DISABLES BOOST CONVERTER WHEN BUS GETS TOO HIGH */

SW2 = (DRIVE_OK & VBOOST_OK & PWM_BOOST);

SW3 = (DRIVE_OK & FORW & PL_HEA & !PL_HEB & PWM_BUCK) #
     (DRIVE_OK & !FORW & !PL_HEA & PL_HEB & PWM_BUCK);

SW4 = (DRIVE_OK & FORW & !PL_HEA & PL_HF9) #
     (DRIVE_OK & !FORW & PL_HEA & !PL_HEB);

SW5 = (DRIVE_OK & FORW & PL_HEB & !PL_HEC & PWM_BUCK) #
     (DRIVE_OK & !FORW & !PL_HEB & PL_HEC & PWM_BUCK);

SW6 = (DRIVE_OK & FORW & !PL_HEB & PL_HEC) #
     (DRIVE_OK & !FORW & PL_HEB & !PL_HEC);

SW7 = (DRIVE_OK & FORW & !PL_HEA & PL_HEC & PWM_BUCK) #
(DRIVE_OK & !FORW & PL_HEA & !PL_HEC & PWM_BUCK);

SW8 = (DRIVE_OK & FORW & PL_HEA & !PL_HEC) #
      (DRIVE_OK & !FORW & !PL_HEA & PL_HEC);

FAULT = !NO_FAULT;

=================================

Chip Diagram

=================================

<table>
<thead>
<tr>
<th></th>
<th>IGBT_SW2</th>
</tr>
</thead>
<tbody>
<tr>
<td>INH</td>
<td>1</td>
</tr>
<tr>
<td>NOT_USED</td>
<td>2</td>
</tr>
<tr>
<td>FORW</td>
<td>3</td>
</tr>
<tr>
<td>MOT</td>
<td>4</td>
</tr>
<tr>
<td>PL_HEA</td>
<td>5</td>
</tr>
<tr>
<td>PL_HEB</td>
<td>6</td>
</tr>
<tr>
<td>PL_HEC</td>
<td>7</td>
</tr>
<tr>
<td>BUCK</td>
<td>8</td>
</tr>
<tr>
<td>PWM_BUCK</td>
<td>9</td>
</tr>
<tr>
<td>PWM_BOOST</td>
<td>10</td>
</tr>
<tr>
<td>VBOOST_OK</td>
<td>11</td>
</tr>
<tr>
<td>GND</td>
<td>12</td>
</tr>
</tbody>
</table>

190
B.3 R-S Latches for User Interface

An eight-button keypad is used to set the drive mode of the motor as shown in Figure 5-19.

With these buttons, the user can select a torque feedback loop or a speed feedback loop\(^1\), set the motor’s direction of rotation, engage or disengage the motor from the drive, and start or stop the motor. This GAL implements the R-S latches used to latch user keypad commands.

```c
Name switches;
Partno CA0006;
Date 11/01/95;
Revision 01;
Designer ofori;
Company MIT;
Assembly None;
Location None;
Device P22V10;

/***************************************************************************/
/* Keypad switches combinational logic (RS latches) */
/***************************************************************************/
/** Inputs **/
PIN 1 = CLK; /* Register Clock */
PIN 2 = TORQUE; /* Torque Mode */
PIN 3 = SPEED; /* Speed Mode */
PIN 4 = FORWARD; /* Spin in forward direction */
PIN 5 = REVERSE; /* Spin in reverse direction */
PIN 6 = DRIVE; /* Inverter drives motor */
PIN 7 = COAST; /* Inverter lets motor coast */
PIN 8 = START; /* Start motor */
PIN 9 = STOP; /* Stop motor */
PIN 10 = INIT; /* Initialize switch settings */

/** Outputs **/
PIN 22 = TORQUE_OUT; /* Debounced torque signal */
PIN 23 = SPEED_OUT; /* Debounced speed signal */
PIN 20 = FORW_OUT; /* Debounced forward signal */
PIN 21 = REV_OUT; /* Debounced reverse signal */
PIN 18 = DRIVE_OUT; /* Debounced drive signal */
PIN 19 = COAST_OUT; /* Debounced coast signal */
PIN 16 = START_OUT; /* Debounced start signal */
```

\(^1\)The speed feedback loop was not implemented in our prototype but the signal is available for use.
PIN 17 = STOP_OUT; /* Debounced stop signal */

/** Logic Equations **/
TORQUE_OUT = !(TORQUE # INIT # SPEED_OUT);
SPEED_OUT = !(TORQUE_OUT # SPEED);
FORW_OUT = !(FORWARD # INIT # REV_OUT);
REV_OUT = !(REVERSE # FORW_OUT);
DRIVE_OUT = !(DRIVE # COAST_OUT);
COAST_OUT = !(COAST # INIT # DRIVE_OUT);
START_OUT = !(START # STOP_OUT);
STOP_OUT = !(STOP # INIT # START_OUT);

==================================================================
Chip Diagram
==================================================================

<table>
<thead>
<tr>
<th>switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK x---</td>
</tr>
<tr>
<td>TORQUE x---</td>
</tr>
<tr>
<td>SPEED x---</td>
</tr>
<tr>
<td>FORWARD x---</td>
</tr>
<tr>
<td>REVERSE x---</td>
</tr>
<tr>
<td>&quot;DRIVE x---</td>
</tr>
<tr>
<td>COAST x---</td>
</tr>
<tr>
<td>START x---</td>
</tr>
<tr>
<td>STOP x---</td>
</tr>
<tr>
<td>INIT x---</td>
</tr>
<tr>
<td>x---</td>
</tr>
<tr>
<td>GND x---</td>
</tr>
</tbody>
</table>

B.4 Miscellaneous Logic

This GAL is used to perform several miscellaneous glue logic. It is used to generate the proper ENABLE pulse for the LCD since the timing requirements for the LCD slightly differ from the bus control signals available from the MC68HC16Z1. The chip is to
buffer the microcontroller's READ/WRITE and ADDRESS 0 signals as well as generate the PWM signals for the buck and boost converters. Finally, the chip is used to generate motor over-current and boost inductor over-current signals. Two versions of this GAL were implemented; one for when the boost converter is switched closed-loop with current-mode control and the other when it is switched open-loop. The difference in these two GALs lie in how the PWM clocks to the IGBTs are generated.

B.4.1 Current-Mode Control of Boost Converter

Name misc_1;
Partno CA0006 ;
Date 11/28/95;
Revision 01;
Designer ofor1;
Company MIT;
Assembly None;
Location None;
Device P22V10;

/*******************************************************************************/
/* LCD enable pulse generator and Boost & Buck SR-Latches */
/*******************************************************************************/

/** Inputs **/
PIN 1 = R_W_IN;  /* Read/Write signal from micro */
PIN 2 = CS3;     /* Chip Select 3 from micro */
PIN 3 = CS3_DELAYED;    /* Chip Select 3 from micro delayed*/
PIN 4 = A0_IN;  /* Address 0 */
PIN 5 = PWMA;  /* Boost converter clock */
PIN 6 = PWMB;  /* Buck converter clock */
PIN 7 = MOT;   /* Motoring signal from micro */
PIN 8 = BOOST_COMP;  /* Output of boost comparator */
PIN 9 = BUCK_COMP;  /* Output of buck comparator */
PIN 10 = CMC;   /* Current mode control mode from micro */
PIN 11 = I_LB_IN_RANGE; /* Boost current in range */
PIN 12 = I_PHASE_IN_RANGE; /* Phase current in range */

/** Outputs **/
PIN 23 = LCD_ENABLE;  /* Chip select for LCD */
PIN 22 = LCD_ENABLE_BAR;  /* Inverted LCD enable to enable HC244 */
PIN 21 = R_W_OUT;   /* Buffered Read/Write signal to LCD */
PIN 20 = A0_OUT;    /* Buffered A0 signal to LCD */
PIN 19 = PWM_BOOST; /* PWM signal for boost converter */
PIN 18 = PWM_BUCK; /* PWM signal for buck converter */
PIN 17 = BOOST_Q;  /* Output of Boost SR Latch */

193
PIN 16 = BUCK_Q;    /* Output of Buck SR Latch */
PIN 15 = I_PH_OVER_RANGE;    /* High when phase current exceeds range */
PIN 14 = I_LB_OVER_RANGE;    /* High when I_LB exceeds range */

/** Declarations and Intermediate Variable Definitions **/
I_LB_RESET =
(BOOST_COMP & !MOT) # (!BOOST_COMP & MOT) # !I_LB_IN_RANGE;

I_PHASE_RESET =
(BUCK_COMP & !MOT) # (!BUCK_COMP & MOT) # !I_PHASE_IN_RANGE;

BOOST_Q = !(PWMA # BOOST_QBAR);

BOOST_QBAR = !(BCOST_Q # I_LB_RESET);

BUCK_Q = !(PWMB # BUCK_QBAR);

BUCK_QBAR = !(BUCK_Q # I_PHASE_RESET);

/** Logic Equations **/
LCD_ENABLE = (!CS3 & !CS3_DELAYED);

LCD_ENABLE_BAR = !LCD_ENABLE;

PWM_BOOST = (!CMC & PWMA) # (CMC & BOOST_QBAR);

PWM_BUCK = (!CMC & PWMB) # (CMC & BUCK_QBAR);

A0_OUT = A0_IN;

R_W_OUT = R_W_IN;

I_PH_OVER_RANGE = !I_PHASE_IN_RANGE;

I_LB_OVER_RANGE = !I_LB_IN_RANGE;
### B.4.2 Duty-Cycle Control of Boost Converter

Name  misc_2;
Partno CA0006 ;
Date  11/28/95;
Revision 01;
Designer ofori;
Company MIT;
Assembly None;
Location None;
Device P22V10;

/*****************************/
/* LCD enable pulse generator and Boost & Buck SR-Latches */
/* */
/*****************************/

/** Inputs **/
PIN 1 = R_W_IN;  /* Read/Write signal from micro */
PIN 2 = CS3;  /* Chip Select 3 from micro */
PIN 3 = CS3_DELAYED;  /* Chip Select 3 from micro delayed*/
PIN 4 = A0_IN;  /* Address 0 */
PIN 5 = PWMA;  /* Boost converter clock */
PIN 6 = PWMB;  /* Buck converter clock */
PIN 7 = MOT;  /* Motoring signal from micro */
PIN 8 = BOOST_COMP;  /* Output of boost comparator */
PIN 9 = BUCK_COMP;  /* Output of buck comparator */
PIN 10 = CMC;  /* Current mode control mode from micro */
PIN 11 = I_LB_IN_RANGE;  /* Boost current in range */
PIN 13 = I_PHASE_IN_RANGE;  /* Phase current in range */

/** Outputs **/
PIN 23 = LCD_ENABLE;  /* Chip select for LCD */
PIN 22 = LCD_ENABLE_BAR;  /* Inverted LCD enable to enable HC244 */
PIN 21 = R_W_OUT;  /* Buffered Read/Write signal to LCD */
PIN 20 = A0_OUT;  /* Buffered A0 signal to LCD */
PIN 19 = PWM_BOOST;  /* PWM signal for boost converter */
PIN 18 = PWM_BUCK;  /* PWM signal for buck converter */
PIN 17 = BOOST_Q;  /* Output of Boost SR Latch */
PIN 16 = BUCK_Q;  /* Output of Buck SR Latch */
PIN 15 = I_PH_OVER_RANGE;  /* High when phase current exceeds range */
PIN 14 = I_LB_OVER_RANGE;  /* High when I_LB exceeds range */

/** Declarations and Intermediate Variable Definitions **/
I_LB_RESET =
(BOOST_COMP & !MOT) # (!BOOST_COMP & MOT) # !I_LB_IN_RANGE;

I_PHASE_RESET =
(BUCK_COMP & !MOT) # (!BUCK_COMP & MOT) # !I_PHASE_IN_RANGE;

BOOST_Q = !(PWMA # BOOST_QBAR);

BOOST_QBAR = !(BOOST_Q # I_LB_RESET);

BUCK_Q = !(PWMB # BUCK_QBAR);

BUCK_QBAR = !(BUCK_Q # I_PHASE_RESET);

/** Logic Equations **/
LCD_ENABLE = (!CS3 & !CS3_DELAYED);

LCD_ENABLE_BAR = !LCD_ENABLE;

PWM_BOOST = PWMA;

PWM_BUCK = PWMB;

A0_OUT = A0_IN;

R_W_OUT = R_W_IN;

I_PH_OVER_RANGE = !I_PHASE_IN_RANGE;

I_LB_OVER_RANGE = !I_LB_IN_RANGE;
Chip Diagram

<table>
<thead>
<tr>
<th>misc_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_W_IN x---</td>
</tr>
<tr>
<td>CS3 x---</td>
</tr>
<tr>
<td>CS3 DELAYED x---</td>
</tr>
<tr>
<td>A0_IN x---</td>
</tr>
<tr>
<td>PWMA x---</td>
</tr>
<tr>
<td>PWMB x---</td>
</tr>
<tr>
<td>MOT x---</td>
</tr>
<tr>
<td>BOOST_COMP x---</td>
</tr>
<tr>
<td>BUCK_COMP x---</td>
</tr>
<tr>
<td>CMC x---</td>
</tr>
<tr>
<td>I_LB_IN_RANGE x---</td>
</tr>
<tr>
<td>GND x---</td>
</tr>
<tr>
<td>--------</td>
</tr>
</tbody>
</table>
Appendix C

Determination of Feedback Gains

The feedback gains for the buck and boost converter control loops were determined with Matlab by using the scripts below the allow the closed loop poles to be placed anywhere in the z-plane as mentioned in Sections 5.3 and 6.4.

C.1 Buck Converter Feedback Gains

% This file contains a Matlab script used in determining
% feedback gains for the buck and boost control loops

% The buck control loop

% Lm is the motor’s synchronous inductance
% Rm is the motor resistance
% Ts is the switching frequency

Lm = 4e-3;
Rm = 0.47;
Ts = 244e-6;

% Compute the continuous-time system and input coupling
% matrices

A = [-Rm/Lm];
B = [1/Lm];

% Compute discrete-time model of plant

I = [1];
F = expm(A*Ts);
G = inv(A)*((F-I)*B);

% Form the discrete-time matrices that include the
% integrator and one-sample delay

M = [F 0 G ; Ts 1 0 ; 0 0 0];
N = [0 ; 0 ; 1];

% Find feedback gains using pole placement
% [ki, kz, ku]
% ku = 0;
K = place(M,N,[0.4938 + 0.2447i 0.4938 - 0.2447i 0.9532 ])


C.2 Boost Converter Feedback Gains

% The boost control loop

% Cdc is the boost output capacitor + snubber capacitors
% Lm is the motor's synchronous inductance
% Rm is the motor resistance
% Ts is the switching frequency

Cdc = 48e-6;
Lm = 4e-3;
Rm = 0.47;
Ts = 244e-6;

% Compute the continuous-time system and input coupling
% matrices

A = [0 -1/Cdc ; 1/Lm -Rm/Lm];
B = [1/Cdc ; 0];

% Compute discrete-time model of plant

I = [1 0 ; 0 1];
F = expm(A*Ts);
G = inv(A)*((F-I)*B);

% Form the discrete-time matrices that include the
% integrator and one-sample delay

M = [F [0 ; 0] G ; 0 Ts 1 0 ; 0 0 0 0];
N = [0 ; 0 ; 0 ; 1];

% Find feedback gains using pole placement
% [kv, ki, kz, ku]
K = place(M,N,[0.2 0.4+0.2i 0.4-0.2i 0.9])
Appendix D

Maple Source Code

The magnetic circuit analyses and other analytic expressions that guided our study of Halbach and conventional arrays in Chapter 2, the design of the slotted wheel motor in Chapter 3, and and the inverter in Chapter 5 were done using the symbolic computations package Maple V. The source code included in this chapter show how the analytic expressions used in making design decisions and plotting selected graphs are computed.

D.1 Halbach vs. Conventional Torque Comparison

# This file contains the Maple code used to study the torque production # characteristics of Halbach and conventional arrays.

# include libraries
readlib(optimize):
readlib(isolate):
with(plots):

# definitions of operators. Laplacian, Divergence, Gradient & Curl
lap := proc(x) (1/r)*diff(r*diff(x,r),r)+(1/r^2)*diff(diff(x,phi),phi) + diff(diff(x,z),z) end;
div := proc(x) (1/r)*diff(r*x[1],r)+(1/r)*diff(x[2],phi) + diff(x[3],z) end;
grad := proc(x) [diff(x,r),(1/r)*diff(x,phi),diff(x,z)] end;
curl := proc(x) [simplify(diff(x[3],phi)/r - diff(x[2],z)),
    simplify(diff(x[1],z) - diff(x[3],r)),
    simplify((diff(r*x[2],r) - diff(x[1],phi))/r)] end;

# Solve general magnetic circuit with/without permeable rotor back-iron
print('Radial MAGNETS');
print('az => azimuthal, r => radial');

# Define magnetic scalar potentials for various regions
# Above magnets
psi_ar := C1r*((r/Rri)^k*N) - (Rri/r)^k*N)*sin(k*N*(phi-theta));
# Inside magnets
psi_cr := (C2r*r^k*N) - C3r*r^(-k*N)*sin(k*N*(phi-theta));
# Below magnets (in airgap and cu)
psi_dr := C4r * ((r/Rso)^k*N) - (Rso/r)^k*N)*sin(k*N*(phi-theta));

# Check if scalar potentials are Laplacian
print('Check if Laplace’s equation is satisfied');
simplify(lap(psi_ar));
simplify(lap(psi_cr));
simplify(lap(psi_dr));
Har := grad(-psi_ar); 
Hcr := grad(-psi_cr); 
Hdr := grad(-psi_dr); 
Bar_r := u0*Har[1]; 
Bcr_r := u0*(Hcr[1]+Mkr); 
Bdr_r := u0*Hdr[1]; 
Baaz_r := u0*Har[2]; 
Bcaz_r := u0*Hcr[2]; 
Bdaz_r := u0*Hdr[2];

# Compute surface charges to be used for checks
sigma_smbr := u0*subs(r=R1,Mkr); 
sigma_smcr := -u0*subs(r=Rm,Mkr); 
Mkr := -Mr*(Rm/r)*sin(k*N*(phi-theta)); 
print('Check if divergence of M is zero');
divM := div([Mkr,0,0]);

# Set up equations to be solved.
print('Using the boundary conditions at r = Rm, scalar potential and normal B is continuous');
eqn1 := subs(r=R1,psi_ar)=subs(r=R1,psi_cr):
eqn2 := subs(r=Rm,psi_cr)=subs(r=Rm,psi_dr):
eqn3 := subs(r=R1,Bar_r)=subs(r=R1,Bcr_r)= 0:
eqn4 := subs(r=Rm,Bcr_r)=subs(r=Rm,Bdr_r)= 0:

# Solve for the unknown constants C1r,C2r,C3r,C4r
constants := solve([eqn1,eqn2,eqn3,eqn4],(C1r,C2r,C3r,C4r)):
assign(constants); 

C1r := simplify(expand(C1r));
C2r := simplify(expand(C2r));
C3r := simplify(expand(C3r));
C4r := simplify(expand(C4r));
# Double check solutions
print('Check that boundary conditions are satisfied');

r := Rri;
'Potential' := simplify(psi_ar);
'Jump in tangential H' := simplify(-Har[2]);
r := Rl;
'Jump in normal B' := simplify(expand(simplify(Bar_r - Bcr_r)));
r := Rm;
'Jump in normal B' := simplify(Bcr_r - Bdr_r);
r := Rso;
'Potential' := simplify(psi_dr);
'Jump in tangential H' := simplify(Hdr[2]);

r := 'r';
print(' ');

# Now carry out the same procedure above for the azimuthal magnets
print('Azimuthal MAGNETS');

# Define magnetic scalar potentials for various regions
# Above magnets #
psi_aaz := C1az * ((r/Rri)^(k*N) - (Rri/r)^(k*N))*sin(k*N*(phi-theta));
# Inside magnets
psi_caz := (Rm*Maz/(k*N) + C2az * r^(k*N) - C3az * r^(-k*N))
            *sin(k*N*(phi-theta));
# Below magnets (in airgap and cu
psi_daz := C4az * ((r/Rso)^(k*N) - (Rso/r)^(k*N))sin(k*N*(phi-theta));

# Check if scalar potentials above and below magnets are Laplacian
print('Check if Laplace's equation is satisfied');
simplify(lap(psi_aaz));
simplify(lap(psi_daz));

# Check if potentials within magnets satisfy Poisson's equation
print(' ');
print('Check if Poisson's equation is satisfied for region c');
simplify(lap(psi_caz));
print(' ');

# Define the magnetic fields in the various regions
Haaz := grad(-psi_aaz);
Hcaz := grad(-psi_caz);
Hdaaz := grad(-psi_daz);
Bar_az := u0*Haaz[1];
Bcr_az := u0*Hcaz[1];
Bdr_az := u0*Hdaaz[1];
Baaz_az := u0*Haaz[2];
Bcaz_az := u0*(Hcaz[2]+Mkaz);
Bdaz_az := u0*Hdaz[2];

# Expression for the azimuthal magnetization
Mkaz := (Rm/r)*Maz*cos(k*N*(phi-theta));

# Expression for the volume charge in the azimuthal magnets
rho_m := -u0*div([0,Mkaz,0]);

# Set up the system of equations to be solved.
eqn1 := subs(r=R1,psi_caz)=subs(r=R1,psi_aaz);
eqn2 := subs(r=Rm,psi_caz)=subs(r=Rm,psi_daz);
eqn3 := subs(r=R1,Bcr_az)-subs(r=R1,Bar_az)= 0;
eqn4 := subs(r=Rm,Bcr_az)-subs(r=Rm,Bdr_az)= 0;

# Solve for the unknown arbitrary constants C1az,C3az,C4az,C2az.
constants := solve({eqn1,eqn2,eqn3,eqn4},{C1az,C3az,C4az,C2az});
assign(constants);
C1az := simplify(expand(C1az));
C2az := simplify(expand(C2az));
C3az := simplify(expand(C3az));
C4az := simplify(expand(C4az));
C5az := simplify(expand(C5az));

# Double check the solutions
print('Check that boundary conditions are satisfied');
r := Rri;
'Potential' := simplify(psi_aaz);
'Jump in tangential H' := simplify(expand(u0*Haaz[2]));
r := R1;
'Jump in normal B' := simplify(expand(simplify(expand(Bar_az-
    Bcr_az))));
'Jump in tangential H' := simplify(expand(simplify(expand(u0*Haaz[2]-
    u0*Hcaz[2])))));
r := Rm;
'Jump in normal B' := simplify(expand(simplify(expand(Bcr_az-
    Bdr_az))));
'Jump in tangential H' := simplify(expand(simplify(expand(u0*Hcaz[2]-
    u0*Hdaz[2])))));
r := Rso;
'Potential' := simplify(psi_daz);
'Jump in tangential H' := simplify(expand(simplify(expand(Hdaz[2])))ahi);
r := 'r';
print(' '); print(' Now add the radial and azimuthal potentials');
print('Now calculate the total fields');
psia_radial := psi_ar;
psic_radial := psi_cr;
psid_radial := psi_dr;
psia_azi := psi_aaz;
psic_azi := psi_caz;
psid_azi := psi_daz;

#temporary constants
C1r1 := C1r:
C1r := 'C1r':
C2r1 := C2r:
C2r := 'C2r':
C3r1 := C3r:
C3r := 'C3r':
C4r1 := C4r:
C4r := 'C4r':

Claz1 := Claz:
Claz := 'Claz':
C2az1 := C2az:
C2az := 'C2az':
C3az1 := C3az:
C3az := 'C3az':
C4az1 := C4az:
C4az := 'C4az':
C5az1 := C5az:
C5az := 'C5az':

# Magnetic flux density in the air above the rotor steel
B_amag := [simplify(Bar_r + Bar_azi), simplify(Baaz_r + Baaz_azi)]:

# Magnetic flux density in the magnet
B_cmag := [simplify(Bcr_r + Bcr_azi), simplify(Bcaaz_r + Bcaaz_azi)]:

# Magnetic flux density below the magnet
B_dmag := [simplify(Bdr_r + Bdr_azi), simplify(Bdaaz_r + Bdaaz_azi)]:

phi0 := Pi/(2*N);

# Now calculate the Torque integral
# Integral of the fundamental component of the torque contribution
# from the
# radial magnets
Bdr_norm_int := subs(cos(k*Pi/2)=0,pk_fac*simplify((-2*N*Weff*J0)
*int(int(simplify(r*r*Bdr_r),phi= simplify(phi0-
((Pi/(2*N))*Gamma))
..simplify(phi0+((Pi/(2*N))*Gamma)),r=Rso..Rcu)));

# Commutation angle
theta_com := Pi/(N*Nphases);

# Integral of the fundamental component of the torque contribution
# from the azimuthal magnets
Bdr_horz_int := subs(cos(k*Pi/2)=0,pk_fac*simplify((-2*N*Weff*J0) *int(int(simplify(r^r*Bdr_az),phi= simplify(phi0- ((Pi/(2*N))*Gamma)) ..simplify(phi0+((Pi/(2*N))*Gamma))),r=Rso..Rcu));

# Total fundamental component of torque
Bdr_tot_int := Bdr_norm_int + Bdr_horz_int;
Bdr_int := collect(simplify(Bdr_tot_int),[J0,N,Weff,pk_fac,u0],factor);

# average torque over commutation interval
ave_Bdr_int := int(Bdr_int,theta=evalf(-theta_com/2)..
 evalf(theta_com/2))/theta_com;

# This is the maximum current density under fixed power dissipation
J0max := sqrt((Pohm*sigma)/(Pi*(Rcu^2-Rso^2)*pk_fac*w*Gamma));

# total axial length of motor
w := 0.1279;
# Inner radius of rotor steel
R1 := 0.196
# Outer radius of copper
Rcu := Rm-AG;
# NdFeB magnets will be used
# Magnet remanent flux density at 20 deg. C
Br0 := 1.2;
# Magnet coercive force at 20 deg. C
Hci0 := 700000;
t0 := 20;
# Temperature coefficient of magnet coercive force
Hc_coeff := -0.65;
# Temperature coefficient of magnet remanent flux density
Br_coeff := -0.11;
# Density of magnet
Denmag := 7500;
# Magnet temperature in deg. C
mag_temp := 110;
# Motor running clearance
AG := 0.000625;
# Inner radius of magnets
Rm := sqrt(R1^2 - (m_magnets/(Denmag*Pi*magnet_length * (beta+delta*(1-beta)))));
# Magnet thickness
tmag := R1-Rm;
# Fringing factor
fringe_fac := 0.1;
end_turn_length := Pi*((Rcu+Rso)/(2*N))* tan(Lambda);
Lambda := Pi/6;
magnet_length := (1+fringe_fac)*Weff;
Weff := 0.06:
total_axial_length := magnet_length + 2*end_turn_length:
# 2 out of 3 phases are on at a time
Gamma := 2/3;
# Number of phases
Nphases := 3;
# Resistivity of copper
restCu := (1/(5*10^7))*(234.5+temperature)/(234.5+20);
# Conductivity of copper
sigma := 1/restCu;
# Copper temperature
temperature := 110;
# Copper packing factor
pk_fac := 0.7;
# Ohmic power dissipation in windings
Pohm := 2500;
# Angle of alignment for peak torque within a commutation period
phi0 := Pi/(2*N);
# Outside radius of stator steel
Rso := Rcu-tcu;
# Inner radius of stator steel
Rsi := Rso-tst;

# Include temperature dependence on magnetization
Mo := Br0*(1+(mag_temp-t0)*Br_coeff/100)/u0;
# Incline temperature dependence on coercive force
Hc0 := Hci0*(1+(mag_temp-t0)*Hc_coeff/100);
# Permeability of free space
u0 := evalf(4*Pi*10^(-7));

# Fourier coefficients of radial and azimuthal magnetizations
Mr := (-1)^(k-1)/2*4*Mo*sin(k*beta*Pi/2)/(k*Pi);
Maz := delta *4*Mo*sin(k*alpha*Pi/2)/(k*Pi);

# add only odd harmonics
k := 2*n+1;
# Restore arbitrary constants
C1r := C1r1:
C2r := C2r1:
C3r := C3r1:
C4r := C4r1:
C5r := C5r1:
C1az := C1az1:
C2az := C2az1:
C3az := C3az1:
C4az := C4az1:
C5az := C5az1:

# Free up memory used for the temporary constants
C1r1 := 'C1r1';
C2r1 := 'C2r1';
C3r1 := 'C3r1';
C4r1 := 'C4r1';
C5r1 := 'C5r1';
C1azl := 'C1azl';
C2azl := 'C2azl';
C3azl := 'C3azl';
C4azl := 'C4azl';
C5azl := 'C5azl';

# Compute total radial no load airgap magnetic flux from
# adding five harmonics

norm_noload_flux := evalf(subs(n=0,B_dmag[1])+subs(n=1,B_dmag[1])+
sub(n=2,B_dmag[1])+subs(n=3,B_dmag[1])+
sub(n=4,B_dmag[1])+subs(n=5,B_dmag[1]));

# Compute total azimuthal no load airgap magnetic flux from
# adding five harmonics

azim_noload_flux := evalf(subs(n=0,B_dmag[2])+subs(n=1,B_dmag[2])+
sub(n=2,B_dmag[2])+subs(n=3,B_dmag[2])+
sub(n=4,B_dmag[2])+subs(n=5,B_dmag[2]));

# Finish calculating the torque by adding five harmonics

tmp := evalf(Bdr_int);
Torque := evalf(subs(n=0,tmp)+subs(n=1,tmp)+subs(n=2,tmp)+
sub(n=3,tmp)+subs(n=4,tmp)+subs(n=5,tmp));

# Compute torque per current density
Torque_per_J := Torque/J0;

# Compute average torque over a commutation interval

tmp := evalf(ave_Bdr_int);
ave_Torque := evalf(subs(n=0,tmp)+subs(n=1,tmp)+subs(n=2,tmp)+
sub(n=3,tmp)+subs(n=4,tmp)+subs(n=5,tmp));

# Set the current density to maximum
J0 := evalf(J0max);
theta := 0;
break;

# Plot graphs
# 3d graph for peak Halbach torque with magnet mass and
# pole-arc to pole-pitch ratio (beta)
plot3d((evalf(subs((N=10,delta=1,Rri=evalf(R1)),Torque)),
evalf(subs((N=10,delta=0,Rri=evalf(R1)),Torque))),
beta=0.4..1,m_magnets=0.000001..17,labelfont=[TIMES,BOLD,16],
axesfont=[TIMES,BOLD,16]);

# 3D graph for average Halbach torque with magnet mass and
# pole-arc to pole-pitch ratio (beta)
plot3d({evalf(subs({N=10,delta=1,Rri=10},ave_Torque))},
    evalf(subs({N=10,delta=0,Rri=10},ave_Torque))),
    beta=0.4..1,m_magnets=0.000001..17,
    labelfont=[TIMES,BOLD,16],axesfont=[TIMES,BOLD,16]);

# Graphs for conventional rotor magnet arrangement design
delta := 0;
Rri := evalf(R1);
m_magnets := 16;
N := 10;
beta := 0.75;
tcu := 0.02;

# Graph for variation of current density with copper thickness
plot({evalf(J0max),tcu=0.001..0.02,labelfont=[TIMES,BOLD,16],
    axesfont=[TIMES,BOLD,16]);

delta := 0;
Rri := evalf(R1);
m_magnets := 16;
N := 10;
beta := 0.75;
tcu := 0.02;
plot3d({evalf(norm_noload_flux),phi=0..evalf(2*Pi/N),
    r=evalf(Rso)..evalf(Rcu),labelfont=[TIMES,BOLD,16],
    axesfont=[TIMES,BOLD,16]);

delta := 0;
Rri := evalf(R1);
m_magnets := 16;
N := 10;
beta := 0.75;
tcu := 0.02;
plot3d({evalf(azim_noload_flux),phi=0..evalf(2*Pi/N),
    r=evalf(Rso)..evalf(Rcu),labelfont=[TIMES,BOLD,16],
    axesfont=[TIMES,BOLD,16]);

delta := 1;
Rri := evalf(R1);
m_magnets := 16;
N := 10;
beta := 0.75;
tcu := 0.02;
plot3d({evalf(norm_noload_flux),phi=0..evalf(2*Pi/N),
    r=evalf(Rso)..evalf(Rcu),labelfont=[TIMES,BOLD,16],
    axesfont=[TIMES,BOLD,16]);

# Graphs for Halbach no-load magnetic flux

delta := 1;
Ri := evalf(R1);
m_magnets := 16;
N := 10;
beta := 0.75;
tcu := 0.02;
plot3d(evalf(azim_noload_flux),phi=0..evalf(2*Pi/N),
       r=evalf(Rso)..evalf(Rcu),labelfont=[TIMES,BOLD,16],
       axesfont=[TIMES,BOLD,16]);

D.2 Slotted Motor Design

# This file contains the Maple code used to design the
# conventional slotted wheel motor in Chapter 3.

readlib(optimize):
readlib(piecewise):
readlib(optimize):
readlib(isolate):
readlib(unassign):
with(plots):
alias(I=I,j=sqrt(-1));

# definitions of operators. Laplacian, Divergence, Gradient & Curl
lap := proc(x) (1/r)*diff(r*diff(x,r),r)+(1/r^2)*diff(diff(x,phi),phi)
       + diff(diff(x,z),z) end;

div := proc(x) (1/r)*diff(r*x[1],r)+(1/r)*diff(x[2],phi)
        + diff(x[3],z) end;

grad := proc(x) [diff(x,r),(1/r)*diff(x,phi),diff(x,z)] end;

curl := proc(x) [simplify(diff(x[3],phi)/r - diff(x[2],z)),
        simplify(diff(x[1],z) - diff(x[3],r)),
        simplify((diff(r*x[2],r) - diff(x[1],phi))/r)] end;

# Solve magnetic circuit. Rotor and stator back iron are assumed
# to have infinite permeability.

print('Radial MAGNETS');
#inside magnets
psia := C1 * ((r/Rri)^k*N - (Rri/r)^k*N)*sin(k*N*(phi-theta));

#below magnets (in airgap)
psib := (C2*r^k*N - C3*r^(-k*N))*sin(k*N*(phi-theta));

# in the stator teeth
psic := C4 * ((r/Rso)^k*N - (Rso/r)^k*N)*sin(k*N*(phi-theta));

print('Check if Laplace’s equation is satisfied');
simplify(lap(psia));
simplify(lap(psib));
simplify(lap(psic));

# Define magnetic field intensities
Ha_mag := grad(-psia);
Hb_mag := grad(-psib);
Hc_mag := grad(-psic);

# Define magnetic flux densities
Ba_mag := [u0*(Ha_mag[1]+Mkr),u0*(Ha_mag[2])];
Bb_mag := [u0*Hb_mag[1],u0*Hb_mag[2]];
Bc_mag := [u_teeth*Hc_mag[1], u_teeth*Hc_mag[2]];

Mkr := -Mr*(Rm/r)*sin(k*N*(phi-theta));

# Harmonic contribution of magnet flux in stator steel
# assume no leakage across slots
Phi_stat_mag_k := subs({r=Rcu, theta=0},int(r*Bb_mag[1],
phi=0..Pi/(2*N)));

# Harmonic contribution of magnet flux in rotor steel
Phi_rot_mag_k := subs({r=Rri, theta=0},int(r*Ba_mag[1],
phi=0..Pi/(2*N)));

# Collect 1/3 of the flux from a magnet
Bteeth_mag_k := evalf(int(subs({theta=0, r=Rcu},Bb_mag[1]),
phi=0..evalf(Pi/(3*N)))/evalf(Pi/(3*N)));
print('Check if divergence of M is zero');

\[ \text{divM} := \text{div([Mkr,0,0])}; \]

print('Using the boundary conditions - psi and normal B is continuous');

\[ \text{eqn1} := \text{subs}(r=Rm,\text{psia})=\text{subs}(r=Rm,\text{psib}); \]

\[ \text{eqn2} := \text{subs}(r=Rm,\text{Ba_mag[1]})-\text{subs}(r=Rm,\text{Bb_mag[1]})=0; \]

\[ \text{eqn3} := \text{subs}(r=Rcu\_cart,\text{Bb_mag[1]})-\text{subs}(r=Rcu\_cart,\text{Bc_mag[1]})=0; \]

\[ \text{eqn4} := \text{subs}(r=Rcu\_cart,\text{psib})-\text{subs}(r=Rcu\_cart,\text{psic})=0; \]

\[ \text{constants} := \text{solve([eqn1,eqn2,eqn3,eqn4]),(C1,C2,C3,C4)}; \]

\[ \text{assign(constants)}; \]

\[ \text{C1} := \text{simplify(C1)}; \]

\[ \text{#simplify(collect(expand(C1r),[Rcu,Rm],distributed))}; \]

\[ \text{C2} := \text{simplify(C2)}; \]

\[ \text{C3} := \text{simplify(C3)}; \]

\[ \text{#simplify(collect(expand(C3r),[Rri,Rm,Rri],distributed))}; \]

\[ \text{C4} := \text{simplify(C4)}; \]

print('Check that boundary conditions are satisfied');

\[ r := Rri; \]

'Potential' := simplify(\text{psia});

'Jump in tangential H' := simplify(-\text{Ha_mag}[2]);

\[ r := Rm; \]

'Jump in normal B' := simplify(\text{Ba_mag[1]} - \text{Bb_mag[1]}); 

'Jump in tangential H' := simplify(\text{Ha_mag}[2]-\text{Hb_mag}[2]);

\[ r := Rcu\_cart; \]

'Jump in normal B' := simplify(\text{Bb_mag[1]} - \text{Bc_mag[1]}); 

'Jump in tangential H' := simplify(\text{Hb_mag}[2]-\text{Hc_mag}[2]);

\[ r := Rso; \]

'Potential' := simplify(\text{psic});

'Jump in tangential H' := simplify(\text{Hc_mag}[2]);
# Airgap extended by 5% from Carter coefficient

\[ R_{cu\_cart} := R_{cu}(1-0.05AG); \]
\[ r := 'r'; \]

\[ \text{print(' ');} \]

\[ M_r := (-1)^{(k-1)/2}4M_0\sin(k\beta\pi/2)/(k\pi); \]
\[ k := 2n+1; \]

\[ B_{\text{teeth\_mag}} := \text{evalf(sum(B\text{teeth\_mag\_k},n=0..10)} * \]
\[ \quad (R_{cu}/(r-\delta R_{so})); \]

# magnetic flux density in the stator steel due to magnets
\[ B_{\text{stat\_mag}} := \text{evalf(sum(Phi\_stat\_mag\_k,n=0..10)/t\_stator);} \]

# magnetic flux density in the rotor steel due to magnets
\[ B_{\text{rot\_mag}} := \text{evalf(sum(evalf(Phi\_rot\_mag\_k),n=0..10)/t\_rotor);} \]

\[ \text{print('Armature reaction fields');} \]

# in the magnet and airgap
\[ H_{ga} := Hg\_arm * R_{cu}/r; \]

# in the stator teeth
\[ H_{ta} := Ht\_arm * R_{so}/r; \]

# in the copper slot
\[ H_{cua} := Hcu\_arm * (r-R_{so})/(t_{cu}); \]

# in the slot opening
\[ H_{oa} := Ho\_arm; \]

\[ B_{ta} := u_{\text{teeth}}H_{ta}; \]
\[ B_{ga} := u0H_{ga}; \]
eq1 := 2*Ht_arm*Rso*ln(Rcu/Rso) + 2*Hg_arm*Rcu*ln(Rri/Rcu)
     = 2*J0*A_cu;

eq2 := 2*Ht_arm*Rso*ln((Rso+tcu)/Rso) + 2*Hcu_arm*w_cu +
     2*u0*Hcu_arm*w_tt/(u_teeth) = 2*J0*A_cu;

eq3 := 2*Ht_arm*Rso*ln(Rcu/Rso) + 2*Ho_arm*w_o +
     2*u0*Ho_arm*l_tp/u_teeth = 2*J0*A_cu;

eq4 := u0*Hcu_arm*tcu/2 + u0*Ho_arm*t_o + u0*Hg_arm*l_tp_star
     = u_teeth*Ht_arm*w_tb;

#eq4 := u0*Hcu_arm*tcu/2 + u0*((Hcu_arm+Ho_arm)/2)*xi +
     u0*Ho_arm*t_o + u0*Hg_arm*l_tp = u_teeth*Ht_arm*w_tb;

equations := (eq1, eq2, eq3, eq4);
unknowns := (Ht_arm, Hg_arm, Hcu_arm, Ho_arm);
constants := solve(equations, unknowns);
assign(constants);

# approximate teeth flux with flux at 1/3 distance up base of slot
B_teeth := evalf(subs(r=(Rso+tcu/3), evalf(abs(Bteeth_mag)+ abs(Bta))));
#B_teeth := subs(r=Rcu, evalf(abs(Bteeth_mag)));

w_tb := (2*Pi*Rso/(2*N*3))*(1-delta);
w_cu := (2*Pi*Rso/(2*N*3))* delta;
l_tp := (2*Pi*Rcu/(2*N*3)) - w_o;
l_tp_star := (2*Pi*Rcu/(2*N*3)) - w_cu;
A_cu := w_cu * tcu * k_pf;
w_tt := (2*Pi*(Rso+tcu)/(2*N*3)) - w_cu;
w_o := 0.00309;
xi := 0.577*(w_cu-w_o)/2;
t_o := 0.00266-xi;

rotor_length := (2*Pi*(Rri+(t_rotor/2)))/(2*N);

stator_length := (2*Pi*(Rso-(tst/2)))/(2*N);
mu_rotor := ur;
mu_stator := us;
Torque := evalf(2*N*u0*int(r*Hg_arm, r=Rm..Rri)*(2*Mo*Weff));
### Torque

\[
\text{Torque} = 2Nu0H_{gap} Rcu t_{mag}(2Mo)W_{eff}
\]

### B_stator

\[
\text{B}_{stator} = \text{evalf}(\sqrt{((\text{subs}(r=Rcu, \text{evalf}(-Bgm*(\beta Pi*Rcu/(2N))))^2 + (u_{teeth}Ht_{arm}w_{tb})^2)/t_{stator}})
\]

### B_rotor

\[
\text{B}_{rotor} = \text{evalf}(\sqrt{((\text{subs}(r=Rri, \text{evalf}(-Bmm*(\beta Pi*Rri/(2N))))^2 + (u0Hg_{arm}l_{tp})^2)/t_{rotor}})
\]

### Eddy current and hysteresis losses in the steel.

\[
\text{(1kg = 0.4536 lbs)}
\]

\[
\text{specific stat backiron loss} := (Ke*(Nf)^2*Bstat_mag^2 + Kh*(Nf)*abs(Bstat_mag)^a)/0.4536
\]

\[
\text{specific stat teeth loss} := (Ke*(Nf)^2*Bteeth^2 + Kh*(Nf)*abs(Bteeth)^a)/0.4536
\]

\[
\text{specific rot steel loss} := (Ke*(Nf)^2*B_rotor^2 + Kh*(Nf)*abs(B_rotor)^a)/0.4536
\]

\[
f := \omega/(2\pi)
\]

### Use M19, 29 Gage steel

\[
Ke := 3.218274654*10^{-5};
Kh := 1.066431820363*10^{-2};
a := 1.7928;
\]

\[
\text{stator teeth loss} := \text{mass stator teeth*specific stat teeth loss}:
\]

\[
\text{stator backiron loss} := \text{mass stator backiron*specific stat backiron loss}:
\]

\[
\text{stator steel loss} := \text{stator teeth loss + stator backiron loss}:
\]

### Calculate motor efficiency

\[
\text{Ohmic power} := \text{evalf}((J0^2*(2*2N^A_{cu}*
(\text{Wohm}+2*\text{ohmic end turn length}))/\sigma)/\sigma)
\]

\[
\text{mech power out} := \text{evalf(Torque*omega)}
\]

\[
\text{motor losses} := \text{evalf(Ohmic power+stator steel loss+ripple power)}
\]

\[
\text{tot power in} := \text{mech power out + motor losses}
\]

\[
\text{motor efficiency} := \text{evalf(mech power out/(mech power out+motor losses))}
\]

\[
J0_{max} := \text{evalf(sqrt((Pohm_max*\sigma)/(2*2N^A_{cu}*
\]

215
(Wohm+2*ohmic_end_turn_length));
Rm := sqrt(Rri^2 - (m_magnets/(Denmag*beta*Pi*magnet_length)));
tmag := Rri-Rm;
end_turn_length := (Pi/(2*N))*((Rcu+Rso)/2)* (1-delta)/
(sqrt(1-(1-delta)^2));
ohmic_end_turn_length := end_turn_length/delta;
magnet_length := Weff;
#Weff := w-2*end_turn_length:
total_axial_length := magnet_length + 2*end_turn_length;
Nphases := 3;
restCu := (1/(5*10^7))*(234.5+temperature)/(234.5+20);
sigma := 1/restCu;
temperature := 120;
k_pf := 0.7;
Rso := Rcu-t_o-xi-tcu;
Rsi := Rso-t_stator;

Rri := R0-tRotor;
Rcu := Rm-AG;
skew_angle := arctan(evalf(((Rso+Rcu)/2)*tan(Pi/N/Nphases)/Weff));
Wohm := evalf(Weff/cos(skew_angle));

Mo := Br0*(1+(mag_temp-t0)*Br_coeff/100)/u0;
Hc0 := Hci0*(1+(mag_temp-t0)*Hc_coeff/100);
u0 := evalf(4*Pi*10^(-7));
ur := evalf(500*u0);
us := evalf(500*u0);
mRotor_steel := Densteel*Pi*(R0^2-Rri^2)* Weff:
m_mass_stator_backiron := Densteel*Pi*(Rso^2-Rsi^2)*Weff:
mass_stator_teeth := evalf((Pi*(Rcu^2-Rso^2)-(2*N^3*(w_cu*tcu +
w_o*t_o + (w_cu+w_o)*xi/2)))*Weff*Densteel);

m_stator_steel := mass_stator_backiron + mass_stator_teeth:
m_steel := m_stator_steel+mRotor_steel:
m_Cu := evalf(3*2*N*A_cu*DenCu*(Wohm+2*(end_turn_length))));
Densteel := 7750;
DenCu := 8650;

# M36 steel costs 0.67$/lb and 1 lb = 0.4526 kg
cost_steel := 0.67/0.4536;
cost_Cu := 20;
motor_mass := m_steel + m_magnets + m_Cu;
motor_cost := m_steel * cost_steel + m_magnets * cost_magnet + m_Cu * cost_Cu;

# Motor geometric parameters
# armature wire diameter
# d = 0.0006;

cslot := evalf(m_Cu/(60*2) * cu_spec_heat_cap);
cpole := evalf(mass_stator_teeth/(60*2)*steel_spec_heat_cap);
cback_iron := evalf(mass_stator_backiron/60*steel_spec_heat_cap);
cu_spec_heat_cap := 384;
steel_spec_heat_cap := 502;

# Magnet parameters

Hci0 := 700000;
t0 := 20;
Hc_coeff := -0.65;
Br_coeff := -0.11;
Denmag := 7500;
mag_temp := 110;
cost_magnet := 200;

R0 := 0.206;
AG := 0.000625;
Pohm_max := 2700;
Br0 := 1.2;
beta := 0.9;
t_rotor := 0.01;
t_stator := 0.01;
m_magnets := 1.7;
Weff := 0.06;
N := 10;
tcu := 0.01984;
delta := 0.5;
#J0 := Nturns*I0/A_cu;
u_teeth := 50*u0;
Ri := 0.098;

#Enter B-H Curves

B_tol := 0.01;
u_tol := evalf(0.05*u0);
mu_tol := 0.01;

#B-H curve for M19-29
B_H := [[0,0],[0.6,0.77],[0.84,1.1],[0.9,1.2],[1.02,1.5],[1.14,2],
[1.21,2.5],[1.26,3],[1.32,4],[1.36,5],[1.4,6],[1.42,7],
[1.44,8],[1.5,9],[1.52,16],[1.54,20],[1.58,28],[1.6,32],[1.62,38],[1.66,52],
[1.7,70],[1.76,100],[1.84,170],[1.9,225],[1.95,300],[2.08,1000]]:

# array for storing torque profile with mu
torque_prof := array(1..100):

# array for storing mu
mu_prof := array(1..100):

# B-mu vectors
abs_Bmax := 100;
B_vec := array(1..nops(B_H)+1):
mu_vec := array(1..nops(B_H)+1):

#Now define a few procedures

#Procedure to construct B-mu curve from B-H Curve
#puts results in B_vec[] and mu_vec[]
f_B_mu := proc(flux_vector)
local m, umax;
global B_vec, mu_vec;
B_vec[1] := 0;
umax := mu_vec[1];
for m from 2 by 1 to nops(flux_vector) do
B_vec[m] := flux_vector[m][1];
mu_vec[m] := (flux_vector[m][1]/(flux_vector[m][2]*79.577472));
od;
B_vec[m] := 10;
mu_vec[m] := evalf(u0);
end:

#procedure to find the maximum mu
#result in answer

f_umax := proc(mu_vector)
local m, answer, size;
answer := 0;
size := nops(entries(eval(mu_vector)));
for m from 1 by 1 to size do
if (mu_vector[m] > answer) then
answer := mu_vector[m];
fi;
answer;
od;
end:

#Procedure to find the correct mu given a B.

gemu := proc(B)
local x;
global mu_vec;
x := 1;
if (B >= abs_Bmax) then
u0;
else
while (B_vec[x] <= B) do
    x := x+1;
    od;
if (B_vec[x] = B) then
    mu_vec[x];
else
    mu_vec[x-1] + (B - B_vec[x-1])/(B_vec[x] - B_vec[x-1])
        *(mu_vec[x] - mu_vec[x-1])
fi;
fi;
end:

#Procedure to find the correct mu for teeth.

f_mu_teeth := proc()
global mu_tol, B_tol, B_teeth, u_teeth, u0, temp1, B_teeth simp;
local B_calc, temp, temp_mu, mu_low, mu_high, B_low, B_high;
temp1 := evalf(u_teeth);
u_teeth := 'u_teeth';
B_teeth simp := evalf(B_teeth);
u_teeth := temp1;
B_calc := evalf(B_teeth simp);
temp mu := evalf(u_teeth);
temp := 500;
mu_low := evalf(u0);
mu_high := evalf(5000*u0);
if type(B_calc, numeric) then
    while (temp > mu_tol or temp1 > B_tol) do
        B_calc := evalf(B_teeth simp);
temp mu := getmu(B_calc);
        if temp mu < u_teeth then
            mu_low := max(temp mu, mu_low);
        end if;
u_teeth := evalf((u_teeth+mu_low)/2);
        if B_calc > 1.8 then
            u_teeth := evalf(u_teeth-5*u0*rand()/999999999999);
fi;
else
mu_high := min(temp_mu,mu_high);
u_teeth := evalf((u_teeth+mu_high)/2);
if B_calc > 1.8 then
  u_teeth := evalf(u_teeth+5*u0*rand()/999999999999);
fi;
fi;

temp := evalf(abs(evalf(u_teeth-temp_mu))
            /abs(evalf(u_teeth)));
temp1 := evalf(abs(evalf(B_calc-evalf(B_teeth_simp)))
                /abs(evalf(evalf(B_teeth_simp))));
evalf(u_teeth/u0);
od;
else
  'f_mu_teeth'();
fi;
end;

#procedure to find motor torque for given mass of magnets
f_torque := proc(x)
global f_mu_teeth,Torque,m_magnets;
m_magnets := 'm_magnets';
m_magnets := x;
f_mu_teeth();
evalf(Torque);
end;
#plot(f_torque,0.5..3,numpoints=20);

#procedure to find teeth permeability for given mass of magnets
f_mu := proc(x)
global f_mu_teeth,Torque,m_magnets;
m_magnets := x;
f_mu_teeth();
end;
#procedure to find optimal torque per mass
f_opt_Weff := proc(x)
  global f_mu_teeth, Torque, J0, J0max, Torque, Weff;
  J0 := J0max;
  Weff := x;
  f_mu_teeth();
  evalf(Torque);
end;

#procedure to find torque for a given thickness of copper
f_opt_tcu := proc(x)
  global f_mu_teeth, Torque, tcu, J0, J0max, B_teeth;
  tcu := 'tcu';
  J0 := evalf(J0max);
  tcu := evalf(x);
  f_mu_teeth();
  evalf(Torque);
end;

plot_tcu := proc()
  global f_opt_tcu, J0, J0max, x1, y1, z1;
  x1 := array(1..201);
  y1 := array(1..201);
  z1 := array(1..400);
  J0 := evalf(J0max);
  i := 0;
  for t_cu from 0.005 by evalf((0.03-0.005)/200) to 0.03 do
    i := i+1;
    x1[i] := t_cu;
    y1[i] := f_opt_tcu(t_cu);
    od;
  i := 'i';
  for i from 1 to 200 do
    z1[eval(2*i-1)] := x1[i];
    z1[eval(2*i)] := y1[i];
  od;
end;
plot(eval(z1));
end;

#plot_tcu(eval(z1));
#plot(f_opt_tcu,0.01..0.03);

#procedure to find torque for a given number of pole pairs
f_opt_N := proc(x)
global f_mu_teeth,Torque,tcu,J0,J0max,N;
N := 'N';
J0 := J0max;
N := x;
f_mu_teeth();
evalf(Torque);
end;

#procedure to find how torque changes with ohmic power dissipation
f_torque_prof := proc(x)
global f_mu_teeth,Torque,J0,J0max,Pohm_max;
Pohm_max := 'Pohm_max';
J0 := J0max;
Pohm_max := x;
f_mu_teeth();
evalf(Torque);
end;
#plot(f_torque_prof,1..2500);

#procedure to find how torque changes with maximum radius
f_max_radius := proc(x)
global f_mu_teeth,Torque,J0,J0max,R0;
J0 := J0max;
R0 := x;
f_mu_teeth();
evalf(Torque);
end;
#plot(f_max_radius,0.15..0.19);
# Now construct B-mu curve and store max mu in umax

f_B_mu(B_H):
    umax := f_umax(mu_vec):
    # plot the B_H and B_mu curves
    size := nops(B_H):
    for j from 1 to (2*size) do
        if type(j, odd) then
            'B_H plot vector'[j] := B_H[(j+1)/2][2]*79.577472;
            'B_mu plot vector'[j] := mu_vec[(j+1)/2]/u0;
            'B_H log plot vector'[j] := B_H[(j+1)/2][1];
            'B_mu log plot vector'[j] := B_H[(j+1)/2][1];
        else
            'B_H plot vector'[j] := B_H[j/2][1];
            'B_mu plot vector'[j] := B_H[j/2][1];
            'B_H log plot vector'[j] := B_H[j/2][2]*79.577472;
            'B_mu log plot vector'[j] := mu_vec[j/2]/u0;
        fi;
    od;
    'B_H plot vector' := ['B_H plot vector'[g] $
g=1..(2*size)]:
    'B_mu plot vector' := ['B_mu plot vector'[g] $
g=1..(2*size)]:
    'B_H log plot vector' := ['B_H log plot vector'[g] $
g=1..(2*size)]:
    'B_mu log plot vector' := ['B_mu log plot vector'[g] $
g=1..(2*size)]:

# J0 := J0max;
min_N := 10:
max_N := 15:
r := 'r';
n := 'n';

# deg. C per minute
temp_rise_rate := evalf(60*Pohm_max/((m_steel+m_Cu)*Cp));
# Specific heat capacity of copper

\[ C_p := 384; \]

\[ \text{max\_profile} := 0.21; \]

\[ J_0 := J_0\text{max}; \]

\[ \text{f\_mu\_teeth}(); \]

\[ \text{readlib(piecewise)}; \]

\[ \text{readlib(optimize)}; \]

\[ \text{readlib(isolate)}; \]

\[ \text{readlib(unassign)}; \]

\[ \text{with(plot)}; \]

\[ \text{alias(I=I, j=sqr(-1))}; \]

\[ \text{ripple\_power} := 100; \]

# Input fuds cycle [speed, torque, seconds]

\[ \text{fuds\_cycle} := \{ \{[0, -30, 5], [0, -20, 6], [0, -10, 14], [0, 0, 256], [0, 0, \ldots], \}

\[ [0, 20, 9], [0, 30, 6], [0, 40, 4], [0, 50, 6], [0, 60, 1] \}, \]

\[ \{ [1000, -30, 23], [1000, -20, 9], [1000, -10, 7], [1000, 0, 3], [1000, 10, 1], \}

\[ [1000, 20, 6], [1000, 30, 4], [1000, 40, 3], [1000, 50, 25], [1000, 60, 1] \}, \]

\[ \{ [2000, -30, 16], [2000, -20, 16], [2000, -10, 4], [2000, 0, 9], [2000, 10, 5], \}

\[ [2000, 20, 7], [2000, 30, 8], [2000, 40, 7], [2000, 50, 19], [2000, 60, 0] \}, \]

\[ \{ [3000, -30, 16], [3000, -20, 18], [3000, -10, 15], [3000, 0, 26], [3000, 10, 15], \}

\[ [3000, 20, 24], [3000, 30, 24], [3000, 40, 7], [3000, 50, 9], [3000, 60, 0] \}, \]

\[ \{ [4000, -30, 8], [4000, -20, 8], [4000, -10, 34], [4000, 0, 104], [4000, 10, 113], \}

\[ [4000, 20, 44], [4000, 30, 14], [4000, 40, 5], [4000, 50, 4], [4000, 60, 0] \}, \]

\[ \{ [5000, -30, 3], [5000, -20, 5], [5000, -10, 8], [5000, 0, 104], [5000, 10, 67], \}

\[ [5000, 20, 14], [5000, 30, 5], [5000, 40, 1], [5000, 50, 2], [5000, 60, 0] \}, \]

\[ \{ [6000, -30, 0], [6000, -20, 1], [6000, -10, 7], [6000, 0, 29], [6000, 10, 35], \}

\[ [6000, 20, 7], [6000, 30, 1], [6000, 40, 1], [6000, 50, 0], [6000, 60, 0] \}, \]

\[ \{ [7000, -30, 0], [7000, -20, 0], [7000, -10, 4], [7000, 0, 0], [7000, 10, 0], \}

\[ [7000, 20, 1], [7000, 30, 3], [7000, 40, 0], [7000, 50, 0], [7000, 60, 0] \}, \]

\[ \{ [8000, -30, 0], [8000, -20, 0], [8000, -10, 3], [8000, 0, 6], [8000, 10, 13], \}

\[ [8000, 20, 5], [8000, 30, 1], [8000, 40, 0], [8000, 50, 0], [8000, 60, 0] \}, \]

\[ \{ [9000, -30, 0], [9000, -20, 0], [9000, -10, 0], [9000, 0, 19], [9000, 10, 29], \}

\[ [9000, 20, 11], [9000, 30, 0], [9000, 40, 0], [9000, 50, 0], [9000, 60, 0] \}, \]
highway_cycle := [[[0, -30, 0], [0, -20, 0], [0, -10, 0], [0, 0, 6], [0, 10, 0],
[0, 20, 0], [0, 30, 1], [0, 40, 0], [0, 50, 0], [0, 60, 0]],
[[1000, -30, 0], [1000, -20, 2], [1000, -10, 1], [1000, 0, 0], [1000, 10, 0],
[1000, 20, 0], [1000, 30, 0], [1000, 40, 0], [1000, 50, 2], [1000, 60, 0]],
[[2000, -30, 0], [2000, -20, 2], [2000, -10, 0], [2000, 0, 0], [2000, 10, 0],
[2000, 20, 0], [2000, 30, 0], [2000, 40, 0], [2000, 50, 2], [2000, 60, 0]],
[[3000, -30, 0], [3000, -20, 2], [3000, -10, 0], [3000, 0, 0], [3000, 10, 0],
[3000, 20, 0], [3000, 30, 0], [3000, 40, 2], [3000, 50, 0], [3000, 60, 0]],
[[4000, -30, 0], [4000, -20, 3], [4000, -10, 0], [4000, 0, 0], [4000, 10, 0],
[4000, 20, 0], [4000, 30, 1], [4000, 40, 2], [4000, 50, 0], [4000, 60, 0]],
[[5000, -30, 1], [5000, -20, 1], [5000, -10, 2], [5000, 0, 3], [5000, 10, 2],
[5000, 20, 7], [5000, 30, 1], [5000, 40, 0], [5000, 50, 0], [5000, 60, 0]],
[[6000, -30, 2], [6000, -20, 0], [6000, -10, 2], [6000, 0, 7], [6000, 10, 25],
[6000, 20, 1], [6000, 30, 2], [6000, 40, 1], [6000, 50, 0], [6000, 60, 0]],
[[7000, -30, 0], [7000, -20, 1], [7000, -10, 7], [7000, 0, 19], [7000, 10, 26],
[7000, 20, 20], [7000, 30, 1], [7000, 40, 0], [7000, 50, 0], [7000, 60, 0]],
[[8000, -30, 0], [8000, -20, 1], [8000, -10, 5], [8000, 0, 49], [8000, 10, 177],
[8000, 20, 14], [8000, 30, 0], [8000, 40, 0], [8000, 50, 0], [8000, 60, 0]],
[[9000, -30, 0], [9000, -20, 1], [9000, -20, 1], [9000, 0, 10], [9000, 10, 36], [9000, 10, 112],
[9000, 20, 21], [9000, 30, 0], [9000, 40, 0], [9000, 50, 0], [9000, 60, 0]],
[[10000, -30, 0], [10000, -20, 0], [10000, -10, 0], [10000, 0, 17], [10000, 10, 168],
[10000, 20, 6], [10000, 30, 0], [10000, 40, 0], [10000, 50, 0], [10000, 50, 0]]):

# check if they add up right;
uds_cycle_total := 0;

m := 'm':
n := 'n':

for m from 1 to nops(uds_cycle) do
for n from 1 to nops(uds_cycle[l]) do
uds_cycle_total := uds_cycle_total + uds_cycle[m][n][3];
end do:
end for:
uds_cycle_total;
highway_cycle_total := 0;

m := 'm';

n := 'n';

for m from 1 to nops(highway_cycle) do
    for n from 1 to nops(highway_cycle[1]) do
        highway_cycle_total := highway_cycle_total + highway_cycle[m][n][3];
    od;
od;

highway_cycle_total;

# convert to SI unit & direct drive
m := 'm';

n := 'n';

for m from 1 to nops(fuds_cycle) do
    for n from 1 to nops(fuds_cycle[1]) do
        fuds_cycle_SI[m][n][1] := fuds_cycle[m][n][1]/12.18;
        fuds_cycle_SI[m][n][2] := fuds_cycle[m][n][2]*16.51/4;
        fuds_cycle_SI[m][n][3] := fuds_cycle[m][n][3]/fuds_cycle_total;
        highway_cycle_SI[m][n][1] := highway_cycle[m][n][1]/12.18;
        highway_cycle_SI[m][n][2] := highway_cycle[m][n][2]*16.51/4;
        highway_cycle_SI[m][n][3] := highway_cycle[m][n][3]/highway_cycle_total;
    od;
od;

J0 := 'J0';

max_Torque := evalf(subs({J0=J0max,theta=0},Torque));

# Calculate the Back emf per phase

Area_conductor := A_cu/Nturns;

I0 := simplify(J0max * Area_conductor);

omega := ((2*Pi)/60) * rpm;

max_omega := ((2*Pi)/60) * max_rpm;

max_rpm := 1300;

rpm_corner := evalf(max_rpm/4);

Bemf_const_JcrossB := ((subs(J0 = J0max,Torque)*

227
subs(rpm=rpm_corner,omega)/(2*I0))/rpm_corner:

rpm := 'rpm';
voltage_per_phase_JcrossB := Bemf_const_JcrossB * rpm;
tim := 0;
peak_bemf := 0;
Nturns := 21;
rpm := 'rpm';
tim := 'tim';

Torque_per_J := evalf(Torque/J0);
Torque_per_amp := evalf(Torque/(J0*Area_conductor));
mech_power_out := evalf(subs(J0=peak_torque/
    Torque_per_J,mech_power_out));
tot_power_in := evalf(subs(J0=peak_torque/
    Torque_per_J,tot_power_in));
mot_power_diss := evalf(subs(J0=peak_torque/
    Torque_per_J,motor_losses));

fuds_pow_out := 0;
fuds_pow_in := 0;
highway_pow_out := 0;
highway_pow_in := 0;
m := 'm';
n := 'n';
for m from 1 to nops(fuds_cycle) do
    for n from 1 to nops(fuds_cycle[1]) do
        rpm := fuds_cycle_SI[m][n][1]:
        #prevent division by zero
        V_emf := max(1e-5,evalf(V_emf_base*rpm/rpm_corner));
        peak_torque := max(1e-5,abs(fuds_cycle_SI[m][n][2]));
        fuds_pow_out := evalf(fuds_pow_out +
            fuds_cycle_SI[m][n][3]*mech_power_out);
        fuds_pow_in := evalf(fuds_pow_in +
            fuds_cycle_SI[m][n][3]*tot_power_in);
        highway_pow_out := evalf(highway_pow_out +
            highway_cycle_SI[m][n][3]*mech_power_out);
        highway_pow_in := evalf(highway_pow_in +
highway_cycle_SI[m][n][3]*tot_power_in):

od:

od:

fuds_efficiency := evalf(fuds_pow_out/fuds_pow_in)*100;

highway_efficiency := evalf(highway_pow_out/highway_pow_in)*100;

rpm := 'rpm';

peak_torque := 'peak_torque';

m := 'm';

n := 'n';

V_emf := 'V_emf';

ef := evalf(subs(J0=peak_torque/Torque_per_J,100*motor_efficiency));

Tqmax := evalf(J0max*Torque_per_J);

print('here');

Tq := Torque_per_J;

#FD := plot(subs({rpm=0,t=0},Torque),J0=0..evalf(J0max)
#   ,title = 'Peak Torque');

#FE := plot3d(ef,peak_torque=0..Tqmax,rpm=0..max_rpm,
#   contours=[20,30,40,50,60,70,75,80,85,90,95],
#   view=0..400,labelfont=[HELVETICA,BOLD,10],
#   axesfont=[HELVETICA,BOLD,10],titlefont=[HELVETICA,BOLD,12],
#   labels=['Peak Torque (Nm)','Speed (rpm)','Eff'],
#   title='Efficiency Map of Motor');

F40 := implicitplot(ef=40,rpm=0..max_rpm,peak_torque=0..Tqmax):

F50 := implicitplot(ef=50,rpm=0..max_rpm,peak_torque=0..Tqmax):

F60 := implicitplot(ef=60,rpm=0..max_rpm,peak_torque=0..Tqmax):

F70 := implicitplot(ef=70,rpm=0..max_rpm,peak_torque=0..Tqmax):

F75 := implicitplot(ef=75,rpm=0..max_rpm,peak_torque=0..Tqmax):

F80 := implicitplot(ef=80,rpm=0..max_rpm,peak_torque=0..Tqmax):

F85 := implicitplot(ef=85,rpm=0..max_rpm,peak_torque=0..Tqmax):

F90 := implicitplot(ef=90,rpm=0..max_rpm,peak_torque=0..Tqmax):

F95 := implicitplot(ef=95,rpm=0..max_rpm,peak_torque=0..Tqmax):

max_motor_power := 15000;

tspeed := proc(x)
if x <= rpm_corner then evalf(max_motor_power/(2*PI*rpm_corner/60))
else
evalf(max_motor_power/(2*Pi*x/60));
fi;
end:
env := plot(tspeed,0..max_rpm,linestyle=2):
display({env,F40,F50,F60,F70,F75,F80,F85,F90,F95},
   labelfont=[HELVETICA,BOLD,10],axesfont=[HELVETICA,BOLD,15],
   titlefont=[HELVETICA,BOLD,15],labels=['Speed (rpm)','Peak Torque (Nm)'],title='Efficiency Map of Motor');
break;
#plotsetup(ps,plotoutput='efficiency_map.ps',
   #plotoptions='portrait,noborder');
#plot({env,F20,F30,F40,F50,F60,F70,F75,F80,F85,F90,F95},
   #labelfont=[HELVETICA,BOLD,10],axesfont=[HELVETICA,BOLD,10],
   #titlefont=[HELVETICA,BOLD,12],
   #labels=['Speed (rpm)','Peak Torque (Nm)'],
   #title='Efficiency Map of Motor');

vol_frac := pk_fac;
kcu := 390;
kepox := 1.8;
kst := 20;
kmixl := kepox * ((kepox+kcu)*(kcu+av_sig)-phi1*ksi_2*(kcu-kepox)^2)/
   ((kepox+kcu)*(kepox+av_sig_bar)-phi1*ksi_2*(kcu-kepox)^2);
kmixu := kcu * ((kepox+kcu)*(kepox+av_sig)-phi2*ksi_1*(kcu-kepox)^2)/
   ((kepox+kcu)*(kepox+av_sig_bar)-phi2*ksi_1*(kcu-kepox)^2);

av_sig_bar := kcu*phi1 + kepox*phi2;
av_sig := kepox*phi1 + kcu*phi2;
ksi_2 := phi2/3 - 0.05707*phi2^2;
ksi_1 := 1-ksi_2;
phi2 := vol_frac;
phi1 := 1-phi2;
D.3 Inverter Design

The Maple source code in this section was used to design and analyze the inverter in Chapter 5.

readlib(piecewise);
readlib(optimize);
readlib(isolate);
readlib(unassign):
with(plots):
alias(I=I,j=sqrt(-1));

# Power electronics
# Battery has an open circuit voltage of 336 V
# with 0.3 ohms internal resistance
# Battery’s allowable minimum output voltage
# is 255 V.

VB_full_load := 255;
VB_open_circuit := 336;
internal_resistance := 0.3;
#internal_resistance := 0.000000000003;
# 15 kW motor
max_motor_power := 15000;
# Assume 0.5 ohms motor resistance
R_motor := 0.5;
#R_motor := 0.0000000000001;
#Ohmic power dissipation in motor
Pohm_max := R_motor * I_corner^2;

# Compute maximum battery current allowed per wheelmotor
max_battery_current := evalf((VB_open_circuit -
    VB_full_load)/internal_resistance/4);
# Compute the maximum power that the inverter has to process
max_inverter_power := max_motor_power+Pohm_max+losses;
# Assume 1100 Watts of total inverter losses for now
losses := 1100;
#losses := 0;
# Make base voltage 200 Volts
V_emf_base := 200;
I_corner := evalf(max_motor_power/V_emf_base);
base_voltage := evalf(V_emf_base + R_motor*I_corner);
V_emf_max := 4*V_emf_base;
motor_current_at_max_rpm := max_motor_power/V_emf_max;
#R_motor := Pohm_max/(I_corner^2);
# Maximum steady-state inverter output voltage
V_out_max := V_emf_max + motor_current_at_max_rpm * R_motor;

# For the down converter the switching frequency
# in continuous conduction is related to the
# delta ripple in the inductor current by

f_cont_down := simplify(V_out*(1-V_out/V_in)/(Lm*delta_I_down));

delta_I_down_cont := simplify(solve(f_cont_down = fsw_max,delta_I_down));
I_peak_down_cont := simplify(I_out_ave+delta_I_down_cont/2);
I_valley_down_cont := simplify(I_out_ave-delta_I_down_cont/2);
I_peak_down_disc := evalf(2*I_out_ave*sqrt(subs(delta_I_down =
    2*I_out_ave, f_cont_down)/fsw_max));
I_valley_down_disc := 0;

#continuous and discontinuous modes : 1 => continuous, 0 => discontinuous

mode_down := proc(v_emf,i_out_ave)
    local temp, v_out;
    v_out := v_emf+i_out_ave*R_motor;
    temp := subs(I_out_ave=i_out_ave,V_out=v_out), I_valley_down_cont);
    if temp < 0 then
        0
    else
        1
    fi;
end;

I_Lm_down := proc(v_emf,i_out_ave)
    local t2, ans1, ans2;
    global V_emf,I_out_ave,t1,duty,duty_bar;
    V_emf := v_emf;
    I_out_ave := i_out_ave;
    if evalf(I_valley_down_cont) < 0 then
        t1 := evalf(Lm*I_peak_down_disc/(V_in-V_out));
        duty := evalf(t1/T);
        t2 := t1+evalf(Lm*I_peak_down_disc/(V_out));
        duty_bar := evalf((t2-t1)/T);
        ans1 := piecewise(t <= t1, evalf(I_valley_down_disc+
            (V_in-V_out)/Lm*t), t <= t2, evalf(I_peak_down_disc -
            (V_out)/Lm*(t-t1)),0);
        ans1;
    else
        t1 := evalf(Lm*delta_I_down_cont/(V_in-V_out));
        duty := evalf(t1/T);
        t2 := evalf(1/fsw_max);
        duty_bar := evalf((t2-t1)/T);
        ans2 := piecewise(t <= t1, evalf(I_valley_down_cont +
            ((V_in-V_out)/Lm)*t),t <= t2, evalf(I_peak_down_cont -
            V_out/Lm*(t-t1)),0);
        ans2;
    fi;
end;
I_buscap_down := proc(v_emf, i_out_ave)
local t2, ans1, ans2;
global V_emf, I_out_ave, t1, duty, duty_bar, I_Lm_down;
V_emf := v_emf;
I_out_ave := i_out_ave;
I_Lm_down(V_emf, I_out_ave);

piecewise(t <= t1, evalf(I_in - I_Lm_down(v_emf, I_out_ave)),
          evalf(I_in));
end;

I_buscap_rms_down := proc(v_emf, i_out_ave)
global V_emf, I_out_ave, t1, duty, duty_bar, I_Lm_down, T;
V_emf := v_emf;
I_out_ave := i_out_ave;
I_Lm_down(V_emf, I_out_ave);
evalf(sqrt(int(simplify(((I_in - evalf(I_valley_down_cont +
           ((V_in-V_out)/Lm)*t))^2), t=0..t1)/T +
           int(I_in^2, t=t1..T)/T));
end;

#plot3d('I_buscap_rms_down'(v_emf, i_out_ave), v_emf=1..200,
#        i_out_ave=1..min(300, evalf(15000/v_emf)),
#        labelfont=[HELVETICA, BOLD, 16], axesfont=[HELVETICA, BOLD, 16]);

I_Lm_ripple_rms_down := proc(v_emf, i_out_ave)
local t1, t2, T, ans1, ans2;
global V_emf, I_out_ave;
V_emf := v_emf;
I_out_ave := i_out_ave;
T := evalf(1/fsw_max);

if evalf(I_valley_down_cont) < 0 then
  t1 := evalf(Lm*I_peak_down_disc/(V_in-V_out));
  t2 := evalf(t1+evalf(Lm*I_peak_down_disc/(V_out)));
  ans1 := sqrt(int(simplify((I_valley_down_disc+
               (V_in-V_out)/Lm*t - I_out_ave)^2), t=0..t1)+
             int(simplify((I_peak_down_disc - V_out/Lm*
               (t-t1)-I_out_ave)^2), t=t1..t2))/T);
else
  t1 := evalf(Lm*delta_I_down_cont/(V_in-V_out));
  t2 := T;
  ans2 := sqrt(int(simplify((I_valley_down_cont+(V_in-V_out)/
               Lm*t - I_out_ave)^2), t=0..t1)+
             int(simplify((I_peak_down_cont - (V_out)/
               Lm*(t-t1)-I_out_ave)^2), t=t1..t2))/T);
endif;

#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans1;
else
  t1 := evalf(Lm*delta_I_down_cont/(V_in-V_out));
  t2 := T;
  ans2 := sqrt(int(simplify((I_valley_down_cont+(V_in-V_out)/
               Lm*t - I_out_ave)^2), t=0..t1)+
             int(simplify((I_peak_down_cont - (V_out)/
               Lm*(t-t1)-I_out_ave)^2), t=t1..t2))/T);
  #V_emf := 'V_emf';
  #I_out_ave := 'I_out_ave';
ans2;
fi;
end;

I_Lu_ripple_rms_down := proc(v_emf,i_out_ave)
local t1,t2,T,ans1,ans2;
global V_emf,I_out_ave;
V_emf := v_emf;
I_out_ave := i_out_ave;
T := evalf(1/fsw_max);
if evalf(I_valley_down_cont) < 0 then
t1 := evalf(Lm*I_peak_down_disc/(V_in-V_out));
#t2 := t1+evalf(Lm*I_peak_down_disc/(V_out));
ans1 := sqrt((int(simplify((I_valley_down_disc+
      (V_in-V_out)/Lm*t - I_in)^2),t=0..t1))/T);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans1;
else
  t1 := evalf(Lm*delta_I_down_cont/(V_in-V_out));
  #t2 := T;
  ans2 := sqrt((int(simplify((I_valley_down_cont+
      (V_in-V_out)/Lm*t - I_in)^2),t=0..t1))/T);
  #V_emf := 'V_emf';
  #I_out_ave := 'I_out_ave';
  ans2;
fi;
end;

Motor_power_ripple_down := proc(v_emf,i_out_ave)
evalf(I_Lm_ripple_rms_down(v_emf,i_out_ave)^2*R_motor);
end;

Lu_power_ripple_down := proc(v_emf,i_out_ave)
evalf(I_Lu_ripple_rms_down(v_emf,i_out_ave)^2*R_Lu);
end;

fdown_sw := proc(v_emf,i_out_ave)
fsw_max;
end;

Ipk_down := proc(v_emf,i_out_ave)
local temp, v_out;
v_out := v_emf+i_out_ave*R_motor;
temp := subs({I_out_ave=i_out_ave,V_out=v_out},
    I_valley_down_cont);
if temp < 0 then
  subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out},
    I_peak_down_disc);
else
subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out),
    I_peak_down_cont);
fi;
end;

Energy_pk_down := proc(v_emf,i_out_ave)
    local temp, v_out;
    v_out := v_emf+i_out_ave*R_motor;
    temp := subs({I_out_ave=i_out_ave,V_out=v_out),
                I_valley_down_cont);
    if temp < 0 then
        subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out),
            evalf(I_peak_down_disc^2*Lm/2));
    else
        subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out),
            evalf(I_peak_down_cont^2*Lm/2));
    fi;
end;

buck_boundary := proc(v_emf)
    local temp;
    temp := evalf(v_emf);
    if type(temp,numeric) then
        if temp <= VB_open_circuit then
            evalf(min(I_corner,evalf(subs(V_emf=v_emf,max_curr_down))),
            evalf(max_motor_power/v_emf));
        else
            0
        fi;
    else
        'buck_boundary'(v_emf);
    fi;
end;

clip := proc(x)
    if type(x,numeric) then
        if x < 0 then
            0;
        else
            x;
        fi;
    else
        'clip'(x);
    fi;
end;

# the up converter
f_cont_up := V_in*(1-V_in/V_out)/(Lm*delta_I_up);
# a quadratic dependence of ripple on voltage leads
# to constant switching frequency:

delta_I_up_cont := simplify(solve(f_cont_up = fsw_max, delta_I_up));
I_peak_up_cont := simplify(I_in+delta_I_up_cont/2);
I_valley_up_cont := simplify(I_in-delta_I_up_cont/2);
I_peak_up_disc := evalf(2*I_in*sqrt(subs(delta_I_up =
    2*I_in, f_cont_up)/fsw_max));
I_valley_up_disc := 0;

I_Lu_average_up := proc(v_emf,i_out_ave)
local t1,t2, ans1, ans2;
global V_emf,I_out_ave,duty,duty_bar;
V_emf := v_emf;
I_out_ave := i_out_ave;
if evalf(I_valley_up_cont) < 0 then
    t1 := evalf(Lu*I_peak_up_disc/V_in);
duty := evalf(t1/T);
    t2 := t1+evalf(Lu*I_peak_up_disc/(V_out-V_in));
duty_bar := evalf((t2-t1)/T);
    ans1 := piecewise(t <= t1, evalf(I_valley_up_disc+(V_in)/Lu*t),
        t <= t2, evalf(I_peak_up_disc - (V_out-V_in)/Lu*(t-t1)),0);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans1;
else
    t1 := evalf(Lu*delta_I_up_cont/(V_in));
duty := evalf(t1/T);
    t2 := evalf(1/fsw_max);
duty_bar := evalf((t2-t1)/T);
    ans2 := piecewise(t <= t1, evalf(I_valley_up_cont +((V_in)/Lu)*t),
        t <= t2, evalf(I_peak_up_cont - (V_out-V_in)/Lu*(t-t1)),0);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans2;
fi;
end;

# this assumes zero capacitor voltage and inductor current ripple.

I_Co_average_up := proc(v_emf,i_out_ave)
local t1,t2, ans1, ans2;
global V_emf,I_out_ave,duty,duty_bar;
V_emf := v_emf;
I_out_ave := i_out_ave;
if evalf(I_valley_up_cont) < 0 then
    t1 := evalf(Lu*I_peak_up_disc/V_in);
duty := evalf(t1/T);
    t2 := t1+evalf(Lu*I_peak_up_disc/(V_out-V_in));
duty_bar := evalf((t2-t1)/T);
ans1 := piecewise(t <= t1, I_out_ave, t <= t2, evalf(I_peak_up_disc - (V_out-V_in)/Lu*(t-t1)-I_out_ave), I_out_ave);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans1;
else
t1 := evalf(Lu*delta_I_up_cont/(V_in));
duty := evalf(t1/T);
t2 := evalf(1/fsw_max);
duty_bar := evalf((t2-t1)/T);
ans2 := piecewise(t <= t1, -I_out_ave, t <= t2, evalf(I_peak_up_cont - (V_out-V_in)/Lu*(t-t1)-I_out_ave), -I_out_ave);
ans2;
fi;
end;

I_Lu_average_ripple_rms_up := proc(v_emf,i_out_ave)
local t1,t2,T,ans1,ans2;
global V_emf,I_out_ave;
V_emf := v_emf;
I_out_ave := i_out_ave;
T := evalf(1/fsw_max);
if evalf(I_valley_up_cont) < 0 then
t1 := evalf(Lu*I_peak_up_disc/(V_in));
t2 := evalf(t1+evalf(Lu*I_peak_up_disc/(V_out-V_in)));
ans1 := sqrt((int(simplify((I_valley_up_cont+(V_in)/Lu*t - I_in)^2),
t=t0..t)+ int(simplify((I_peak_up_disc - (V_out-V_in)/Lu*(t-t1)-I_in)^2),t=t1..t2))/T);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans1;
else
t1 := evalf(Lu*delta_I_up_cont/(V_in));
t2 := T;
ans2 := sqrt((int(simplify((I_valley_up_cont+(V_in)/Lu*t - I_in)^2),
t=t0..t)+ int(simplify((I_peak_up_cont - (V_out-V_in)/Lu*(t-t1)-I_in)^2),t=t1..t2))/T);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans2;
fi;
end;

V_Co_average_ripple_up := proc(v_emf,i_out_ave)
local t1,t2,T,ans1,ans2,temp;
global V_emf,I_out_ave,Co;
V_emf := v_emf;
I_out_ave := i_out_ave;
T := evalf(1/fsw_max);
if evalf(I_valley_up_cont) < 0 then
t1 := evalf(Lu*I_peak_up_disc/(V_in));
t2 := evalf(t1+evalf(Lu*I_peak_up_disc/(V_out-V_in)));
temp := simplify(solve(I_peak_up_disc - (V_out-V_in)/
    Lu*(t-t1)-I_out_ave = 0,t));
ans1 := simplify((int(I_out_ave,t=0..t1)+ int(simplify(I_peak_up_disc
    - (V_out-V_in)/Lu*(t-t1)-I_out_ave),t=temp..t2) +
    int(I_out_ave,t=t2..T))/Co);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans1;
else
    t1 := evalf(Lu*delta_I_up_cont/(V_in));
t2 := T;
temp := simplify(solve(I_peak_up_cont -
    (V_out-V_in)/Lu*(t-t1)-I_out_ave=0,t));
ans2 := simplify((int(I_out_ave,t=0..t1)+ int(simplify((I_peak_up_cont -
    (V_out-V_in)/Lu*(t-t1)-I_out_ave)),t=temp..t2))/Co);
#V_emf := 'V_emf';
#I_out_ave := 'I_out_ave';
ans2;
    fi;
end;

I_Co_average_rms_up := proc(v_emf,i_out_ave)
    local t1,t2,T,ans1,ans2;
    global V_emf,I_out_ave;
    V_emf := v_emf;
    I_out_ave := i_out_ave;
    T := evalf(1/fsw_max);
    if evalf(I_valley_up_cont) < 0 then
        t1 := evalf(Lu*I_peak_up_disc/(V_in));
        t2 := evalf(t1+evalf(Lu*I_peak_up_disc/(V_out-V_in)));
        ans1 := sqrt((int(I_out_ave^2,t=0..t1)+ int(simplify((I_peak_up_disc -
            (V_out-V_in)/Lu*(t-t1)-I_out_ave)^2),t=t1..t2) +
            int(I_out_ave^2,t=t2..T))/T));
        #V_emf := 'V_emf';
        #I_out_ave := 'I_out_ave';
        ans1;
        else
            t1 := evalf(Lu*delta_I_up_cont/(V_in));
            t2 := T;
            ans2 := sqrt((int(I_out_ave^2,t=0..t1)+ int(simplify((I_peak_up_cont -
                (V_out-V_in)/Lu*(t-t1)-I_out_ave)^2),t=t1..t2))/T));
            #V_emf := 'V_emf';
            #I_out_ave := 'I_out_ave';
            ans2;
            fi;
        end;
    Lu_power_ripple_up := proc(v_emf,i_out_ave)
evalf(I_Lu_ripple_rms_up(v_emf,i_out_ave)^2*R_Lu);
end;

fup_sw := proc(v_emf,i_out_ave)
  fsw_max;
end;

mode_up := proc(v_emf,i_out_ave)
  local temp, v_out;
  v_out := v_emf+i_out_ave*R_motor;
  temp := subs({I_out_ave=i_out_ave,V_out=v_out}, I_valley_up_cont);
  if temp < 0 then
    0
  else
    1
  fi;
end;

Ipk_up := proc(v_emf,i_out_ave)
  local temp, v_out;
  v_out := v_emf+i_out_ave*R_motor;
  temp := subs({I_out_ave=i_out_ave,V_out=v_out},
              I_valley_up_cont);
  if temp < 0 then
    subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out},
         I_peak_up_disc);
  else
    subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out},
         I_peak_up_cont);
  fi;
end;

Energy_pk_up := proc(v_emf,i_out_ave)
  local temp, v_out;
  v_out := v_emf+i_out_ave*R_motor;
  temp := subs({I_out_ave=i_out_ave,V_out=v_out},
              I_valley_up_cont);
  if temp < 0 then
    subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out},
         evalf(I_peak_up_disc^2*Iu/2));
  else
    subs({I_out_ave=i_out_ave,V_emf = v_emf,V_out=v_out},
         evalf(I_peak_up_cont^2*Iu/2));
  fi;
end;

fsw_max := 16440;
T := evalf(1/fsw_max);
#Lm :=evalf(268e-6 *(V_emf_max/894)^2);
Lm := 8e-3;
#internal_resistance := 1e-15;
Rs := 4*internal_resistance;
#V_out_max := 1000;
#Pohm := 2300;
max_motor_power := 15000;

V_out := V_emf+I_out_ave*R_motor;
V_emf_max := max(solve(subs(I_out_ave = (max_motor_power/V_emf),
                       V_out)=V_out_max,V_emf));

V_in := VB_open_circuit - I_in*Rs;
I_in := VB_open_circuit/(2*Rs) - (VB_open_circuit^2 -
       4*Rs*V_out*I_out_ave)^(1/2)/(2*Rs);
#V_in := 250;
#I_in := V_out*I_out_ave/V_in;
#I_out_ave := I_corner;
#V_emf_down_ct_max := max(evalf(allvalues(solve(V_in=V_out,V_emf))));
#I_out_ave := max_motor_power/V_emf;
#V_emf_down_cp_max := fsolve(V_in=V_out,V_emf=V_emf_base..V_emf_max);
I_out_ave := 'I_out_ave';
max_curr_down := max(evalf(allvalues(solve(V_in=V_out,I_out_ave))));
I_out_ave := 'I_out_ave';

#combined switching frequency

f_sw := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
  fdown_sw(v_emf,i_ave);
else
  fup_sw(v_emf,i_ave);
fi;
end;
mode := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
  mode_down(v_emf,i_ave);
else
  mode_up(v_emf,i_ave);
fi;
end;

I_co_rms_ave := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
  0;
end;
else
I_Co_average_rms_up(v_emf,i_ave);
fi;
end;

V_Co_ripple_ave := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
0;
else
V_Co_average_ripple_up(v_emf,i_ave);
fi;
end;

I_Lu_pk := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
Ipk_down(v_emf,i_ave);
else
Ipk_up(v_emf,i_ave);
fi;
end;

Energy_Lu_pk := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
Energy_pk_down(v_emf,i_ave);
else
Energy_pk_up(v_emf,i_ave);
fi;
end;

Lu_power_ripple := proc(v_emf,i_ave)
local temp;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
Lu_power_ripple_down(v_emf,i_ave);
else
Lu_power_ripple_up(v_emf,i_ave);
fi;
end;

Lu_energy_at_max_rpm := proc(inductance)
global Lu,V_emf,I_out_ave;
Lu := inductance;
V_emf := V_emf_max;
I_out_ave := max_motor_power/V_emf_max;
Energy_pk_up(V_emf,I_out_ave);
end;

Lu_power_ripple_at_max_rpm := proc(inductance)
  global Lu,V_emf,I_out_ave;
  Lu := inductance;
  V_emf := V_emf_max;
  I_out_ave := max_motor_power/V_emf_max;
  Lu_power_ripple_up(V_emf,I_out_ave);
end;

 resistance of inductor; R proportional to N
R_Lu := 0.019;
max_core_loss := 170;
#: 0.01*sqrt(Lu/30e-6);

 Critical maximum inductance for minimum energy
 storage at max. V_emf
V_emf_clip := 1200;
V_emf := V_emf_clip;
I_out_ave := max_motor_power/V_emf;
Lu_crit := V_in^2*(1-V_in/V_out)/(2*max_motor_power*fsw_max);
Lu := Lu_crit;
V_emf := 'V_emf';
I_out_ave := 'I_out_ave';

 Semiconductor parameters

V_d_on := 2.5;
t_on := 0.8e-6;
t_off := 1.5e-6;
tr := 0.6e-6;
tf := 0.5e-6;

 down conversion

P_s1_cond_down := 0;

P_s2_cond_down := proc()
  local Vt2, a2, b2;
  Vt2 := 0.4819;
a2 := 0.1196;
b2 := 0.4881;
evalf((Vt2+a2*I_in^b2)*I_in);
end;

P_s3_s8_cond_down := proc()
  local Vt1, Vt2, a1, a2, b1, b2, I1, I2,
          E_trans, E_diode, E_trans1, E_trans2, t1;
  Vt1 := 1.011;
a1 := 0.2324;
b1 := 0.5085;
Vt2 := 0.4819;
a2 := 0.1196;
b2 := 0.4881;
I1 := subs(t=0, I_Lm_down(V_emf, I_out_ave));
I2 := subs(t=duty*T, I_Lm_down(V_emf, I_out_ave));
E_trans := Vt1*(I1+I2)*duty*T/2 +
abs((I2^2-(b1+2)-I1^2-(b1+2))/(I2-I1))*
((a1*duty*T)/(b1+2));
E_trans1 := E_trans;
I1 := I2;
I2 := subs(t=T, I_Lm_down(V_emf, I_out_ave));
E_diode := Vt2*(I1+I2)*duty_bar*T/2 +
abs((I2^2-(b2+2)-I1^2-(b2+2))/(I2-I1))*
((a2*duty_bar)*T)/(b2+2);
E_trans2 := E_diode*Vt1/Vt2;
evalf((E_trans+E_diode+E_trans1+E_trans2)/T);
end;

#2*V_ce_sat*I_out_ave*duty +2*V_d_on*I_out_ave*duty_bar
P_RLu_cond_down := I_in^2*R_Lu;
#P_s3_s8_sw_down := 2*fsw_max*(E_on_down+E_off_down);

P_s3_s8_sw_down := proc()
local Vt1, Vt2, a1, a2, b1, b2,
I1, I2, E_trans_on, E_trans_off, E_sw, t1;
a2 := 0.0000294;
b2 := 1.3977682;
a1 := 0.0002064516;
b1 := 1;
I1 := subs(t=0, I_Lm_down(V_emf, I_out_ave));
I2 := subs(t=duty*T, I_Lm_down(V_emf, I_out_ave));
E_trans_on := V_in*a1*I1^b1/600;
E_trans_off := V_in*a2*I2^b2/600;
E_sw := E_trans_on + E_trans_off;
evalf(E_sw*fsw_max);
end;

P_RLu_cond_up := proc()
local I1, I2;
I1 := subs(t=0, I_Lu_average_up(V_emf, I_out_ave));
I2 := subs(t=duty*T, I_Lu_average_up(V_emf, I_out_ave));
evalf(R_Lu*(I1^2+I2^2+I1*I2)/3) +
evalf((V_emf*I_out_ave)*max_core_loss/max_motor_power);
end;

tot_pow_diss_down := proc(v_emf, i_ave)
global V_emf, I_out_ave;

local temp;
V_emf := v_emf;
I_out_ave := i_ave;
temp := evalf(P_s2_cond_down()+P_s3_s8_cond_down() + 
P_RLu_cond_down+P_s3_s8_sw_down());
end;

P_s1_s8_cond_up := proc()
local Vt1,Vt2, a1,a2, b1,b2, I1,I2,
E_trans,E_diode,E_12,t1;
Vt1 := 1.011;
a1 := 0.2324;
b1 := 0.5085;
Vt2 := 0.4819;
a2 := 0.1196;
b2 := 0.4881;
I1 := subs(t=0,I_Lu_average_up(V_emf,I_out_ave));
I2 := subs(t=duty*T,I_Lu_average_up(V_emf,I_out_ave));
E_trans := Vt1*(I1+I2)*duty*T/2+abs((I2*(b1+2)-I1*(b1+2))
/(I2-I1))*(a1*duty*T)/(b1+2);
I1 := subs(t=duty*T,I_Lu_average_up(V_emf,I_out_ave));
I2 := subs(t=(duty+duty_bar)*T,I_Lu_average_up(V_emf,I_out_ave));
E_diode := Vt2*(I1+I2)*duty_bar*T/2+
abs((I2*(b2+2)-I1*(b2+2))/(I2-I1))*(a2*duty_bar*T)/(b2+2);
E_12 := evalf((E_trans+E_diode)/T);
evalf(E_12+2*2*Vt1*I_out_ave);
end;

P_s1_s2_sw_down := proc()
local Vt1,Vt2, a1,a2, b1,b2, I1,I2,E_trans_on,
E_trans_off,E_sw,t1;
a2 := 0.0000294;
b2 := 1.3977682;
a1 := 0.0002064516;
b1 := 1;
I1 := subs(t=0,I_Lu_average_up(V_emf,I_out_ave));
I2 := subs(t=duty*T,I_Lu_average_up(V_emf,I_out_ave));
E_trans_on := V_out*a1*I1^b1/600;
E_trans_off := V_out*a2*I2^b2/600;
E_sw := E_trans_on +E_trans_off;
evalf(E_sw*fsw_max);
end;

P_s3_s8_sw_down := proc()
local Vt1,Vt2, a1,a2, b1,b2, I1,I2,E_trans_on,
E_trans_off,E_sw,t1,fcom;
a2 := 0.0000294;
b2 := 1.3977682;
a1 := 0.0002064516;
b1 := 1;
I1 := I_out_ave;
I2 := I_out_ave;
fcom := (V_emf/V_emf_max)*6*max_rpm*N/60;
E_trans_on := V_out*a1*I1^b1/600;
E_trans_off := V_out*a2*I2^b2/600;
E_sw := E_trans_on + E_trans_off;
evalf(E_sw*fcom);
end;

tot_pow_diss_up := proc(v_emf,i_ave)
global V_emf,I_out_ave;
local E_on_up,E_off_up,temp;
V_emf := v_emf;
I_out_ave := i_ave;
temp := evalf(P_s1_s2_sw_up()+P_s1_s8_cond_up())
       +P_RLu_cond_up()+P_s3_s8_sw_up());
end;

tot_power_diss := proc(v_emf,i_ave)
global V_emf,I_out_ave;
local temp;
V_emf := v_emf;
I_out_ave := i_ave;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
tot_pow_diss_down(v_emf,i_ave);
else
tot_pow_diss_up(v_emf,i_ave);
fi;
end;

switching_losses := proc(v_emf,i_ave)
global V_emf,I_out_ave;
local temp;
V_emf := v_emf;
I_out_ave := i_ave;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
evalf(P_s3_s8_sw_down());
else
evalf(P_s1_s2_sw_up()+P_s3_s8_sw_up());
fi;
end;

conduction_losses := proc(v_emf,i_ave)
global V_emf,I_out_ave;
local temp;
V_emf := v_emf;
I_out_ave := i_ave;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
evalf(P_s2_cond_down()+P_s3_s8_cond_down());
else
evalf(P_s1_s8_cond_up());
fi;
end;

inv_power_diss := proc()
local temp;
global I_out_ave;
I_out_ave := evalf(peak_torque/Torque_per_amp);
if type(I_out_ave,numeric) and type(V_emf,numeric) then
temp := buck_boundary(V_emf);
if V_emf <= VB_open_circuit and I_out_ave <= temp then
tot_pow_diss_down(v_emf,I_out_ave);
else
tot_pow_diss_up(V_emf,I_out_ave);
fi;
else
'inv_power_diss'();
fi;
end;

inv_efficiency := proc(v_emf,i_ave)
global V_emf, I_out_ave;
local temp,temp1,temp2,gdrive_power;
V_emf := v_emf;
I_out_ave := i_ave;
gdrive_power := 20;
temp := buck_boundary(v_emf);
if v_emf <= VB_open_circuit and i_ave <= temp then
temp1 := tot_pow_diss_down(v_emf,i_ave);
temp2 := evalf(100*V_out*I_out_ave/(V_in*
        I_in+temp1+gdrive_power));
V_emf := 'V_emf';
I_out_ave := 'I_out_ave';
temp2;
else
temp1 := tot_pow_diss_up(v_emf,i_ave);
temp2 := evalf(100*V_out*I_out_ave/
        (V_in*I_in+temp1+gdrive_power));
V_emf := 'V_emf';
I_out_ave := 'I_out_ave';
temp2;
fi;
end;
power_elec_cost := proc()
local c_switch, c_cap_hv, c_cap_lv, c_ind, c_gdrives;
c_cap_hv := 20;
c_switch := 85.50;
c_gdrives := 5;
c_cap_lv := 50;
evalf(c_ind+4*c_switch+c_cap_lv+c_cap_hv+4*c_gdrives);
end;
break;

display((plot(`conduction_losses`(vemf,40),
vemf = 0.1..min(V_emf_max,evalf(15000/40))),
plot(`conduction_losses`(vemf,50),
vemf = 0.1..min(V_emf_max,evalf(15000/50))),
plot(`conduction_losses`(vemf,18.75),
vemf = 0.1..min(V_emf_max,evalf(15000/18.75))),
plot(`conduction_losses`(vemf,30),
vemf = 0.1..min(V_emf_max,evalf(15000/30))),
plot(`conduction_losses`(vemf,10),
vemf = 0.1..min(V_emf_max,evalf(15000/10))),
plot(`conduction_losses`(vemf,75),
vemf = 0.1..min(V_emf_max,evalf(15000/75))),
plot(`conduction_losses`(vemf,60),
vemf = 0.1..min(V_emf_max,evalf(15000/60)))))
labelfont=[HELVETICA,BOLD,16],axesfont=[HELVETICA,BOLD,16]);

display((plot(`switching_losses`(vemf,40),
vemf = 0.1..min(V_emf_max,evalf(15000/40))),
plot(`switching_losses`(vemf,50),
vemf = 0.1..min(V_emf_max,evalf(15000/50))),
plot(`switching_losses`(vemf,18.75),
vemf = 0.1..min(V_emf_max,evalf(15000/18.75))),
plot(`switching_losses`(vemf,30),
vemf = 0.1..min(V_emf_max,evalf(15000/30))),
plot(`switching_losses`(vemf,10),
vemf = 0.1..min(V_emf_max,evalf(15000/10))),
plot(`switching_losses`(vemf,75),
vemf = 0.1..min(V_emf_max,evalf(15000/75))),
plot(`switching_losses`(vemf,60),
vemf = 0.1..min(V_emf_max,evalf(15000/60)))))
labelfont=[HELVETICA,BOLD,16],axesfont=[HELVETICA,BOLD,16]);

display((plot(`tot_power_diss`(vemf,40),
vemf = 0.1..min(V_emf_max,evalf(15000/40)))),
plot('tot_power_diss'(vemf,50),
  vemf = 0.1..min(V_emf_max,evalf(15000/50)) ),
plot('tot_power_diss'(vemf,18.75),
  vemf = 0.1..min(V_emf_max,evalf(15000/18.75)) ),
plot('tot_power_diss'(vemf,30),
  vemf = 0.1..min(V_emf_max,evalf(15000/30)) ),
plot('tot_power_diss'(vemf,10),
  vemf = 0.1..min(V_emf_max,evalf(15000/10)) ),
plot('tot_power_diss'(vemf,75),
  vemf = 0.1..min(V_emf_max,evalf(15000/75)) ),
plot('tot_power_diss'(vemf,60),
  vemf = 0.1..min(V_emf_max,evalf(15000/60)) ),
labelfont=[HELVETICA, BOLD, 16], axesfont=[HELVETICA, BOLD, 16]);

FA := plot(min(75,evalf(max_motor_power/vemf)),
  vemf=0.001..800,labelfont=[HELVETICA, BOLD, 16],
  axesfont=[HELVETICA, BOLD, 16], linestyle=2):

display(FA, implicitplot('inv_efficiency'(vemf,iout)=90,
  vemf=0.001..800,iout=0.001..75),
  implicitplot('inv_efficiency'(vemf,iout)=93,
  vemf=0.001..800,iout=0.001..75),
  implicitplot('inv_efficiency'(vemf,iout)=95,
  vemf=0.001..800,iout=0.001..75),
  implicitplot('inv_efficiency'(vemf,iout)=96,
  vemf=0.001..800,iout=0.001..75),
  labelfont=[HELVETICA, BOLD, 16], axesfont=[HELVETICA, BOLD, 16]);
Appendix E

Microcode

This appendix contains the assembly language code for prototype inverter and motor control, as described in Chapter 5. Detailed information on the MC68HC16Z1 processor and its instruction set can be found in [60] - [65].

******************************************************************************
*   Title : EQUATES                                                   *
*   Description : This is a table of EQUates for all of the *
*                   registers in the HC16.                          *
*                                                                *
*   Note : This program is written for the M68HC16Z1EVB          *
******************************************************************************

******************************************************************************
**** variables in ram        ****
**** on chip 1k ram begins at f000h        ****
**** for words, high byte is stored first        ****
******************************************************************************

$org       $f000               ; variable address start

*** periodic sampling clock is 244 us *****

ki_pll    equ   $27bd            ; pll int constant  (cutoff at 100 hz)
            ; # = 2*pi*fc * ts * 2^16
ki_dc     equ   $7d8             ; dc filter (cutoff at 20 hz)
ki_low_cutoff equ $3ec           ; low cutoff frequency filter
ki_high_cutoff equ $27bd         ; high cutoff frequency filter

atod_midpoint equ $1fa           ; (vref-vrlp)=5.12v, vref= *****
maxsteps  equ   !20              ; number of steps in commutation period
torque_constant equ !6           ; torque per amp (nm per amp)
corner_rpm equ   !325             ; corner rpm
max_const_torque equ (((161 * $1ff)/(100 )) ; max average torque
            ; (450 nm -- 0.61* $1ff)
max_bus_voltage equ (((1400 * $7fe)/1400) ; max bus voltage
max_boost_voltage equ ($(1200 * $17fa)/1200); max boost voltage
     ; (400 v -- $7fe)
max_phase_current equ ($(100 * $1ff)/100); max phase current
     ; (100 a -- $1ff)
max_boostdac_val equ $2aa
     ; maximum dac value 7ff
;max_boost_current equ ($(200 * $3ff)/200); max boost inductor
     ; current (200 a -- $3ff)
max_boost_current equ ($(1200 * $2fe)/200); max boost inductor
     ; current (200 a -- $3ff)
max_temp equ ($(120 * $3ff)/180); max winding (hot spot) temperature
     ; (180 deg c -- $3ff)
max_rpm equ !1300
     ; maximum motor rpm
max_trq_rpm equ {corner_rpm * max_const_torque}; max product of
     ;torque * corner rpm
max_trpm_hi equ max_trq_rpm/$10000
     ; high byte
max_trpm_lo equ max_trq_rpm%$10000
     ; low byte
k_emf equ $18d
     ; 0.777 volts/rpm
     ; 0.777/200 * $3ff * 1000 (divide by 1000)
rpm_count equ !393185
     ; divide this number by the number of
     ;input capture
     ;counts to get motor rpm
     ; (60 sec/min)/(count * 10 counts/rev * 
     ; 15.26 us/count)
max_shp_rpm equ !350  ; max rpm for torque shaping
min_shp_rpm equ !7
     ; min rpm for torque shaping since timer runs out
min_oc_rpm equ !30
     ; frequency below which we disable hall effect
     ; output comparators
soft_start_rpm equ !1300  ; above this speed we heavily filter
     ;torque command
min_pav_rpm equ !50
     ; speed at which we start phase advance
max_filt_rpm equ !1300
     ; speed above which we filter the input
     ;torque command
temp1 dw $0000h
     ; 2-byte storage for temporary variables
temp2 dw $0000h
     ; temporary storage of pc during display
temp3 dw $0000
     ; temporary storage for boost integrator
temp4 dw $0000
     ; temporary storage for boost integrator
temp_reg dw 0000
     ; temp register. delete later
     ; contains full value of torque pot
max_perm_torque dw $000h
     ; contains maximum permissible torque
mode db 0c2h
     ; control mode
     ; start in torque, forward, coast, stop mode
     ; need to push start and drive for action
mode_chg equ $02
     
flags db %01001111
     ; status flags
tof equ $01
     ; timer overflow
ioca equ $02 ; set when output compare on phase a is inhibited
iocb equ $04 ; set when output compare on phase b is inhibited
iocc equ $08 ; set when output compare on phase c is inhibited
fault equ $10 ; set when the current in the selected phase has
; to be negated for servo
lds equ $20 ; leading zero flag. used to display numbers
; in decimal on lcd
soft_start equ $40 ; soft start flag
good_atod equ $80 ; set if a/d value is good
leading_sign db $00 ; temporary storage for leading sign for
; disnum subroutine

dec_places db $00 ; number of decimal places to be displayed by
; disnum

old_tic1 dw $0000 ; previous ic1 capture time
new_hea_period dw $ffff ; new period of phase a hall effect sensor
hea_periodh dw $ffff ; high byte of period of phase a hall effect
; sensor signal
hea_periodl dw $ffff ; low byte of period of phase a hall effect
; sensor signal

old_tic2 dw $0000 ; previous ic2 capture time
new_heb_period dw $ffff ; new period of phase b hall effect sensor
heb_periodh dw $ffff ; high byte of period of phase b hall effect
; sensor signal
heb_periodl dw $ffff ; low byte of period of phase b hall effect
; sensor signal

old_tic3 dw $0000 ; previous ic3 capture time
new_hec_period dw $ffff ; new period of phase c hall effect sensor
hec_periodh dw $ffff ; high byte of period of phase c hall effect
; sensor signal
hec_periodl dw $ffff ; low byte of period of phase c hall effect
; sensor signal

heabc_period ; average of hea_periodh, heb_periodh, and
; hec_periodh
percent_adv dw $0000 ; percent phase advance
to_count db $02 ; number of times timer overflowed between
; input capture interrupts
comcnt db $00 ; torque ripple compensation pointer

motor_rpm dw $0000 ; motor rpm
bemf dw $0000 ; back emf
bemf_by_tfh dw $0000 ; back emf * ff (high byte)
bemf_by_ffl dw $0000 ; back emf * ff (low byte)

new_bus_volt dw $0000 ; unfiltered bus voltage
dc_bus_voltl dw $0000 ; high byte of filtered dc bus voltage
dc_bus_voltl dw $0000 ; low byte of filtered dc bus voltage
vbus_by_ffh    dw $0000 ; high byte of dc bus voltage multiplied by ff
vbus_by_ffl    dw $0000 ; low word of dc bus voltage multiplied by ff

new_boost_volt dw $0000 ; unfiltered boost voltage
dc_boost_volt h dw $0000 ; high byte of filtered dc boost voltage
dc_boost_volt l dw $0000 ; low byte of filtered dc boost voltage
vboost_by_ffh  dw $0000 ; high byte of dc boost voltage multiplied by ff
vboost_by_ffl  dw $0000 ; low word of dc boost voltage multiplied by ff

new_phase_curr dw $0000 ; unfiltered phase current
dc_phase_curr h dw $0000 ; high byte of filtered dc phase current
dc_phase_curr l dw $0000 ; low byte of filtered dc phase current
iphasa_zerop  dw $0200 ; offset on phasea current channel (non-inverted)
iphasa_zeron  dw $0200 ; offset on phasea current channel (inverted)
iphasb_zerop  dw $0200 ; offset on phaseb current channel (non-inverted)
iphasb_zeron  dw $0200 ; offset on phaseb current channel (inverted)
iphasc_zerop  dw $0200 ; offset on phasc current channel (non-inverted)
iphasc_zeron  dw $0200 ; offset on phasc current channel (inverted)
iphasc_zero   dw $01f3 ; temporary
new_boost_curr dw $0000 ; unfiltered boost current
dc_boost_curr h dw $0000 ; high byte of filtered dc boost current
dc_boost_curr l dw $0000 ; low byte of filtered dc boost current
ilb_zero      dw $0200 ; offset on boost current channel

new_torque_cmd dw $0000 ; instantaneous torque command
fil_torque_cmd h dw $0000 ; high byte of filtered commanded torque
fil_torque_cmd l dw $0000 ; low byte of filtered commanded torque
comp_torque_cmd dw $0000 ; commanded torque, compensated for ; commutation torque ripple
                       ; torque servo is closed around this number
phase_curr_err dw $0000 ; phase current error for torque loop

new_brake_cmd  dw $0000 ; instantaneous brake command
fil_brake_cmd h dw $0000 ; high byte of filtered brake command
fil_brake_cmd l dw $0000 ; low byte of filtered brake command
comp_brake_cmd dw $0000 ;

winding_temp dw $0000 ; winding temperature
endturn_temp  dw $0000 ; end turn temperature
commtime     dw $0000 ; time of last commutaiton

;* control variables (buck, motor)

buckm_interr h dw $0000 ; high byte of integrator error
buckm_interr l dw $0000 ; low byte of integrator error
kp_buck_mot   equ $5fa ; proportional gain
;kp_buck_mot   equ $4fb
ki_buck_mot   equ $38 ; integral gain
ku_buck_mot_sat equ $ffff ; (-4 * 0.266) multiplier of old command in ;buck, motor, saturation mode
ki_buck_mot_sat equ $04 ; (4 * 2 * 0.5267) multiplier of phase current
kv_buck_mot_sat equ $28f5 ; (4 * 0.08 * 7fff) multiplier of boost voltage

;* control variables - boost motor
mult_fac dw $0000 ; contains 2*ts*(vout-vin)/lu
boost_cmd_data dw $0000 ; storage for computed boost dac data
boost_ipk dw $0000 ; peak boost current
ki_boost_mot equ $6500 ; integral gain (0.6344 * $8000)
max_voltage dw $0000 ; max boost voltage
kp_boost_mot equ $1 ;
;ku_boost_mot equ $01 ; command gain
kv_boost_mot equ $6eaa ; voltage gain (-0.237 * $8000)
boostm_interrh dw $0000 ; high byte of integrator error
boostm_interrl dw $0000 ; low byte of integrator error

boost_volt_cmd dw $0000 ; commanded boost voltage from buck controller
pwmdata dw $0000 ; where newly computed pwm data is stored
boostdac_data dw $0000 ; newly computed data for boost dac

motor_resistance dw $3c28 ; 0.47 * $8000

proftab dw $0000 ;

***** sim module registers *****

simmcr equ $fa00 ; sim module configuration register
simtr equ $fa02 ; system integration test register
syncr equ $fa04 ; clock synthesizer control register
rser equ $fa07 ; reset status register
simtre equ $fa08 ; system integration test register (e clock)
porte0 equ $fa11 ; port e data register (same data as portel)
boostled equ $80
buckled equ $40
init equ $20
* neg equ $10 ; has been defined in flags
porte1 equ $fa13 ; port e data register (same data as porte0)
porte equ porte0 ; port e data register (same data as porte0)
ddre equ $fa15 ; port e data direction register
pepar equ $fa17 ; port e pin assignment register
portf0 equ $fa19 ; port f data register (same data as portf1)
strt equ $10
driv equ $20
forw equ $40
torq equ $80
портf1 equ $fa1b ; port f data register (same data as portf0)
portf equ portf0 ; port f data register (same data as portf0)
ddrf equ $fa1d ; port f data direction register
ppfar equ $fa1f ; port f pin assignment register
syspcr equ $fa21 ; system protection control register
picr equ $fa22 ; periodic interrupt control register
pitr equ $fa24 ;periodic interrupt timing register
swsr equ $fa27 ;software service register
tstmsra equ $fa30 ;master shift register a
tstmsrb equ $fa32 ;master shift register b
tstsc equ $fa34 ;test module shift count
tstrc equ $fa36 ;test module repetition count
creg equ $fa38 ;test submodule control register
dreg equ $fa3a ;distributed register
cspdr equ $fa41 ;port c data register
portc equ $fa41 ;port c data register (same as cspdr)
buck equ $40
mot equ $20
*for equ $10
lock equ $08
inh equ $04
cmc equ $02
cspar0 equ $fa44 ;chip-select pin assignment register 0
cspar1 equ $fa46 ;chip-select pin assignment register 1
csbarbt equ $fa48 ;chip-select boot base address register
csorbt equ $fa4a ;chip-select boot option register
csbar0 equ $fa4c ;chip-select 0 base address register
csor0 equ $fa4e ;chip select 0 option register
csbar1 equ $fa50 ;chip-select 1 base address register
csor1 equ $fa52 ;chip-select 1 option register
csbar2 equ $fa54 ;chip-select 2 base address register
csor2 equ $fa56 ;chip-select 2 option register
csbar3 equ $fa58 ;chip-select 3 base address register
csor3 equ $fa5a ;chip-select 3 option register
csbar4 equ $fa5c ;chip-select 4 base address register
csor4 equ $fa5e ;chip-select 4 option register
csbar5 equ $fa60 ;chip-select 5 base address register
csor5 equ $fa62 ;chip-select 5 option register
csbar6 equ $fa64 ;chip-select 6 base address register
csor6 equ $fa66 ;chip-select 6 option register
csbar7 equ $fa68 ;chip-select 7 base address register
csor7 equ $fa6a ;chip-select 7 option register
csbar8 equ $fa6c ;chip-select 8 base address register
csor8 equ $fa6e ;chip-select 8 option register
csbar9 equ $fa70 ;chip-select 9 base address register
csor9 equ $fa72 ;chip-select 9 option register
csbar10 equ $fa74 ;chip-select 10 base address register
csor10 equ $fa76 ;chip-select 10 option register

***** sram module registers *****
rammcr equ $fb00 ;ram module configuration register
ramtsr equ $fb02 ;ram test register
rambah equ $fb04 ;ram base address high register
rambal equ $fb06 ;ram base address low register
***** qsm address map *****

qmcrc equ $fc00 ;qsm module configuration register
qtest equ $fc02 ;qsm test register
qilr equ $fc04 ;qsm interrupt levels register
qivr equ $fc05 ;qsm interrupt vector register
sccr0 equ $fc08 ;sci control register 0
sccr1 equ $fc0a ;sci control register 1
scsr equ $fc0c ;sci status register
scdr equ $fc0e ;sci data register
qdr equ $fc15 ;qsm port data register
qpar equ $fc16 ;qsm pin assignment register
qddr equ $fc17 ;qsm data direction register
spcr0 equ $fc18 ;qspi control register 0
spcr1 equ $fc1a ;qspi control register 1
spcr2 equ $fc1c ;qspi control register 2
spcr3 equ $fc1e ;qspi control register 3
spsr equ $fc1f ;qspi status register
rr0 equ $fd00 ;spi rec.ram 0
rr1 equ $fd02 ;spi rec.ram 1
rr2 equ $fd04 ;spi rec.ram 2
rr3 equ $fd06 ;spi rec.ram 3
rr4 equ $fd08 ;spi rec.ram 4
rr5 equ $fd0a ;spi rec.ram 5
rr6 equ $fd0c ;spi rec.ram 6
rr7 equ $fd0e ;spi rec.ram 7
rr8 equ $fd00 ;spi rec.ram 8
rr9 equ $fd02 ;spi rec.ram 9
rra equ $fd04 ;spi rec.ram a
rrb equ $fd06 ;spi rec.ram b
rrc equ $fd08 ;spi rec.ram c
rrd equ $fd0a ;spi rec.ram d
rre equ $fd0c ;spi rec.ram e
rrf equ $fd0e ;spi rec.ram f
tr0 equ $fd20 ;spi txd.ram 0
tr1 equ $fd22 ;spi txd.ram 1
tr2 equ $fd24 ;spi txd.ram 2
tr3 equ $fd26 ;spi txd.ram 3
tr4 equ $fd28 ;spi txd.ram 4
tr5 equ $fd2a ;spi txd.ram 5
tr6 equ $fd2c ;spi txd.ram 6
tr7 equ $fd2e ;spi txd.ram 7
tr8 equ $fd30 ;spi txd.ram 8
tr9 equ $fd32 ;spi txd.ram 9
tra equ $fd34 ;spi txd.ram a
trb equ $fd36 ;spi txd.ram b
trc equ $fd38 ;spi txd.ram c
trd equ $fd3a ;spi txd.ram d
tre equ $fd3c ;spi txd.ram e

255
trf equ $fd3e ;spi txd.ram f
cr0 equ $fd40 ;spi cmd.ram 0
cr1 equ $fd41 ;spi cmd.ram 1/
cr2 equ $fd42 ;spi cmd.ram 2
cr3 equ $fd43 ;spi cmd.ram 3
cr4 equ $fd44 ;spi cmd.ram 4
cr5 equ $fd45 ;spi cmd.ram 5
cr6 equ $fd46 ;spi cmd.ram 6
cr7 equ $fd47 ;spi cmd.ram 7
cr8 equ $fd48 ;spi cmd.ram 8
cr9 equ $fd49 ;spi cmd.ram 9
cra equ $fd4a ;spi cmd.ram a
crb equ $fd4b ;spi cmd.ram b
crc equ $fd4c ;spi cmd.ram c
crd equ $fd4d ;spi cmd.ram d
cre equ $fd4e ;spi cmd.ram e
crf equ $fd4f ;spi cmd.ram f

***** gpt module registers *****
gptmcr equ $f900 ;gpt module configuration register
gptmtr equ $f902 ;gpt module test register (reserved)
gpticr equ $f904 ;gpt interrupt configuration register
gptpddr equ $f906 ;parallel data direction register
gptpdr equ $f907 ;parallel data register

hea equ $01
heb equ $02
hec equ $04
; pgp3 not used
heap1 equ $10
hebpl equ $20
hecpl equ $40

ocl1m equ $f908 ;ocl action mask register
ocl1d equ $f909 ;ocl action data register
tcnt equ $f90a ;timer counter register
pact1 equ $f90c ;pulse accumulator control register
pactn equ $f90d ;pulse accumulator counter
tic1 equ $f90e ;input capture register 1
tic2 equ $f910 ;input capture register 2
tic3 equ $f912 ;input capture register 3
toc1 equ $f914 ;output compare register 1
toc2 equ $f916 ;output compare register 2
toc3 equ $f918 ;output compare register 3
toc4 equ $f91a ;output compare register 4
ti4o5 equ $f91c ;input capture 4 or output compare 5
tctcl1 equ $f91e ;timer control register 1
tctcl2 equ $f91f ;timer control register 2
tmsk1 equ $f920 ;timer interrupt mask register 1

256
tmsk2 equ $f921 ; timer interrupt mask register 2
tflag1 equ $f922 ; timer interrupt flag register 1
tflag2 equ $f923 ; timer interrupt flag register 2
cforc equ $f924 ; compare force register
pwmc equ $f925 ; pwm control register
pwma equ $f926 ; pwm register a
pwmb equ $f927 ; pwm register b
pwmsregs equ pwma ; so that word storage can be done
pwmcnt equ $f928 ; pwm counter register
pwmbufa equ $f92a ; pwm buffer register a
pwmbufb equ $f92b ; pwm buffer register b
prescl equ $f92c ; gpt prescaler

***** adc module registers *****

adcmcr equ $f700 ; adc module configuration register
adtest equ $f702 ; adc test register
adcddr equ $f706 ; adc port data register
adctl0 equ $f70a ; a/d control register 0
adctl1 equ $f70c ; a/d control register 1
adstat equ $f70e ; adc status register
adstatcfequ adstat+1 ; adc conversion complete flags
iph_f equ $01 ; phase current flag
ilb_f equ $02 ; boost inductor current flag
vbst_f equ $04 ; bus voltage flag
vbus_f equ $08 ; boost voltage flag
torg_f equ $10 ; torque command flag
brk_f equ $20 ; brake command flag
wtemp_f equ $40 ; winding temperature flag
etmp_f equ $80 ; endturn temperature flag
rjurr0 equ $f710 ; right justified unsigned result register 0
rjurr1 equ $f712 ; right justified unsigned result register 1
rjurr2 equ $f714 ; right justified unsigned result register 2
rjurr3 equ $f716 ; right justified unsigned result register 3
rjurr4 equ $f718 ; right justified unsigned result register 4
rjurr5 equ $f71a ; right justified unsigned result register 5
rjurr6 equ $f71c ; right justified unsigned result register 6
rjurr7 equ $f71e ; right justified unsigned result register 7
ljurrd0 equ $f720 ; left justified signed result register 0
ljurrd1 equ $f722 ; left justified signed result register 1
ljurrd2 equ $f724 ; left justified signed result register 2
ljurrd3 equ $f726 ; left justified signed result register 3
ljurrd4 equ $f728 ; left justified signed result register 4
ljurrd5 equ $f72a ; left justified signed result register 5
ljurrd6 equ $f72c ; left justified signed result register 6
ljurrd7 equ $f72e ; left justified signed result register 7
ljurr0 equ $f730 ; left justified unsigned result register 0
ljurr1 equ $f732 ; left justified unsigned result register 1
ljurr2 equ $f734 ; left justified unsigned result register 2
ljurr3 equ $f736 ; left justified unsigned result register 3
ljurr4 equ $f738 ;left justified unsigned result register 4
ljurr5 equ $f73a ;left justified unsigned result register 5
ljurr6 equ $f73c ;left justified unsigned result register 6
ljurr7 equ $f73e ;left justified unsigned result register 7

***** (external) lcd registers *****
instr_reg equ $f800 ;lcd instruction register
data_reg equ $f804 ;lcd data register

*******************************************************************************
* Title : ORG00008
* Description : This file initializes the interrupt/
* exception vectors ($0008 - $01fe).
* If an interrupt occurs requiring the use of
* any of these vectors, program flow will
* continue at the label "bdm" which must be
* added by the programmer to his/her code to
* put the program into background debug mode
* or some other appropriate routine.
*
*******************************************************************************

org $0008 ;put the following code in memory
;starting at address $0008 of the map
;(after the reset vector).
;there is a total of 252 of these
;"dw bdm" lines
;vector number (in base 10)
;and vector description

dw bdm ;4  breakpoint (bkpt)
dw bdm ;5  bus error (berr)
dw bdm ;6  software interrupt (swi)
dw bdm ;7  illegal instruction
dw bdm ;8  divide by zero
dw bdm ;9  (unassigned reserved)
dw bdm ;10 (unassigned reserved)
dw bdm ;11 (unassigned reserved)
dw bdm ;12 (unassigned reserved)
dw bdm ;13 (unassigned reserved)
dw bdm ;14 (unassigned reserved)
dw bdm ;15 uninitialized interrupt
dw bdm ;16 (unassigned reserved)
dw bdm ;17 level 1 interrupt autovector
dw bdm ;18 level 2 interrupt autovector
dw bdm ;19 level 3 interrupt autovector
dw bdm ;20 level 4 interrupt autovector
dw bdm ;21 level 5 interrupt autovector
dw bdm ;22 level 6 interrupt autovector
dw bdm ;23 level 7 interrupt autovector
dw bdm ;24 spurious interrupt
; 25 (unassigned reserved)
.dw bdm ;26 (unassigned reserved)
.dw bdm ;27 (unassigned reserved)
.dw bdm ;28 (unassigned reserved)
.dw bdm ;29 (unassigned reserved)
.dw bdm ;30 (unassigned reserved)
.dw bdm ;31 (unassigned reserved)
.dw bdm ;32 (unassigned reserved)
.dw bdm ;33 (unassigned reserved)
.dw bdm ;34 (unassigned reserved)
.dw bdm ;35 (unassigned reserved)
.dw bdm ;36 (unassigned reserved)
.dw bdm ;37 (unassigned reserved)
.dw bdm ;38 (unassigned reserved)
.dw bdm ;39 (unassigned reserved)
.dw bdm ;40 (unassigned reserved)
.dw bdm ;41 (unassigned reserved)
.dw bdm ;42 (unassigned reserved)
.dw bdm ;43 (unassigned reserved)
.dw bdm ;44 (unassigned reserved)
.dw bdm ;45 (unassigned reserved)
.dw bdm ;46 (unassigned reserved)
.dw bdm ;47 (unassigned reserved)
.dw bdm ;48 (unassigned reserved)
.dw bdm ;49 (unassigned reserved)
.dw bdm ;50 (unassigned reserved)
.dw bdm ;51 (unassigned reserved)
.dw bdm ;52 (unassigned reserved)
.dw bdm ;53 (unassigned reserved)
.dw bdm ;54 (unassigned reserved)
.dw bdm ;55 (unassigned reserved)
.dw bdm ;56 user defined interrupt vector 1
.dw bdm ;57 user defined interrupt vector 2
.dw bdm ;58 user defined interrupt vector 3
.dw bdm ;59 user defined interrupt vector 4
.dw bdm ;60 user defined interrupt vector 5
.dw bdm ;61 user defined interrupt vector 6
.dw bdm ;62 user defined interrupt vector 7
.dw bdm ;63 user defined interrupt vector 8
.dw bdm ;64 user defined interrupt vector 9
.dw bdm ;65 user defined interrupt vector 10
.dw bdm ;66 user defined interrupt vector 11
.dw bdm ;67 user defined interrupt vector 12
.dw bdm ;68 user defined interrupt vector 13
.dw bdm ;69 user defined interrupt vector 14
.dw bdm ;70 user defined interrupt vector 15
.dw bdm ;71 user defined interrupt vector 16
.dw bdm ;72 user defined interrupt vector 17
.dw bdm ;73 user defined interrupt vector 18
.dw bdm ;74 user defined interrupt vector 19
.dw bdm ;75 user defined interrupt vector 20
dw bdm ;76 user defined interrupt vector 21
dw bdm ;77 user defined interrupt vector 22
dw bdm ;78 user defined interrupt vector 23
dw bdm ;79 user defined interrupt vector 24
dw bdm ;80 user defined interrupt vector 25
dw bdm ;81 user defined interrupt vector 26
dw bdm ;82 user defined interrupt vector 27
dw bdm ;83 user defined interrupt vector 28
dw bdm ;84 user defined interrupt vector 29
dw bdm ;85 user defined interrupt vector 30
dw bdm ;86 user defined interrupt vector 31
dw bdm ;87 user defined interrupt vector 32
dw bdm ;88 user defined interrupt vector 33
dw bdm ;89 user defined interrupt vector 34
dw bdm ;90 user defined interrupt vector 35
dw bdm ;91 user defined interrupt vector 36
dw bdm ;92 user defined interrupt vector 37
dw bdm ;93 user defined interrupt vector 38
dw bdm ;94 user defined interrupt vector 39
dw bdm ;95 user defined interrupt vector 40
dw bdm ;96 user defined interrupt vector 41
dw bdm ;97 user defined interrupt vector 42
dw bdm ;98 user defined interrupt vector 43
dw bdm ;99 user defined interrupt vector 44
dw bdm ;100 user defined interrupt vector 45
dw bdm ;101 user defined interrupt vector 46
dw bdm ;102 user defined interrupt vector 47
dw bdm ;103 user defined interrupt vector 48
dw bdm ;104 user defined interrupt vector 49
dw bdm ;105 user defined interrupt vector 50
dw bdm ;106 user defined interrupt vector 51
dw bdm ;107 user defined interrupt vector 52
dw bdm ;108 user defined interrupt vector 53
dw bdm ;109 user defined interrupt vector 54
dw bdm ;110 user defined interrupt vector 55
dw bdm ;111 user defined interrupt vector 56
dw bdm ;112 user defined interrupt vector 57
dw bdm ;113 user defined interrupt vector 58
dw bdm ;114 user defined interrupt vector 59
dw bdm ;115 user defined interrupt vector 60
dw bdm ;116 user defined interrupt vector 61
dw bdm ;117 user defined interrupt vector 62
dw bdm ;118 user defined interrupt vector 63
dw bdm ;119 user defined interrupt vector 64
dw bdm ;120 user defined interrupt vector 65
dw bdm ;121 user defined interrupt vector 66
dw bdm ;122 user defined interrupt vector 67
dw bdm ;123 user defined interrupt vector 68
dw bdm ;124 user defined interrupt vector 69
dw bdm ;125 user defined interrupt vector 70
dw bdm ;126 user defined interrupt vector 71

260
\texttt{dw bdm ;127 user defined interrupt vector 72}
\texttt{dw bdm ;128 user defined interrupt vector 73}
\texttt{dw bdm ;129 user defined interrupt vector 74}
\texttt{dw bdm ;130 user defined interrupt vector 75}
\texttt{dw bdm ;131 user defined interrupt vector 76}
\texttt{dw bdm ;132 user defined interrupt vector 77}
\texttt{dw bdm ;133 user defined interrupt vector 78}
\texttt{dw bdm ;134 user defined interrupt vector 79}
\texttt{dw bdm ;135 user defined interrupt vector 80}
\texttt{dw bdm ;136 user defined interrupt vector 81}
\texttt{dw bdm ;137 user defined interrupt vector 82}
\texttt{dw bdm ;138 user defined interrupt vector 83}
\texttt{dw bdm ;139 user defined interrupt vector 84}
\texttt{dw bdm ;140 user defined interrupt vector 85}
\texttt{dw bdm ;141 user defined interrupt vector 86}
\texttt{dw bdm ;142 user defined interrupt vector 87}
\texttt{dw bdm ;143 user defined interrupt vector 88}
\texttt{dw bdm ;144 user defined interrupt vector 89}
\texttt{dw bdm ;145 user defined interrupt vector 90}
\texttt{dw bdm ;146 user defined interrupt vector 91}
\texttt{dw bdm ;147 user defined interrupt vector 92}
\texttt{dw bdm ;148 user defined interrupt vector 93}
\texttt{dw bdm ;149 user defined interrupt vector 94}
\texttt{dw bdm ;150 user defined interrupt vector 95}
\texttt{dw bdm ;151 user defined interrupt vector 96}
\texttt{dw bdm ;152 user defined interrupt vector 97}
\texttt{dw bdm ;153 user defined interrupt vector 98}
\texttt{dw bdm ;154 user defined interrupt vector 99}
\texttt{dw bdm ;155 user defined interrupt vector 100}
\texttt{dw bdm ;156 user defined interrupt vector 101}
\texttt{dw bdm ;157 user defined interrupt vector 102}
\texttt{dw bdm ;158 user defined interrupt vector 103}
\texttt{dw bdm ;159 user defined interrupt vector 104}
\texttt{dw bdm ;160 user defined interrupt vector 105}
\texttt{dw bdm ;161 user defined interrupt vector 106}
\texttt{dw bdm ;162 user defined interrupt vector 107}
\texttt{dw bdm ;163 user defined interrupt vector 108}
\texttt{dw bdm ;164 user defined interrupt vector 109}
\texttt{dw bdm ;165 user defined interrupt vector 110}
\texttt{dw bdm ;166 user defined interrupt vector 111}
\texttt{dw bdm ;167 user defined interrupt vector 112}
\texttt{dw bdm ;168 user defined interrupt vector 113}
\texttt{dw bdm ;169 user defined interrupt vector 114}
\texttt{dw bdm ;170 user defined interrupt vector 115}
\texttt{dw bdm ;171 user defined interrupt vector 116}
\texttt{dw bdm ;172 user defined interrupt vector 117}
\texttt{dw bdm ;173 user defined interrupt vector 118}
\texttt{dw bdm ;174 user defined interrupt vector 119}
\texttt{dw bdm ;175 user defined interrupt vector 120}
\texttt{dw bdm ;176 user defined interrupt vector 121}
\texttt{dw bdm ;177 user defined interrupt vector 122}
dw bdm ;178 user defined interrupt vector 123
dw bdm ;179 user defined interrupt vector 124
dw bdm ;180 user defined interrupt vector 125
dw bdm ;181 user defined interrupt vector 126
dw bdm ;182 user defined interrupt vector 127
dw bdm ;183 user defined interrupt vector 128
dw bdm ;184 user defined interrupt vector 129
dw bdm ;185 user defined interrupt vector 130
dw bdm ;186 user defined interrupt vector 131
dw bdm ;187 user defined interrupt vector 132
dw bdm ;188 user defined interrupt vector 133
dw bdm ;189 user defined interrupt vector 134
dw bdm ;190 user defined interrupt vector 135
dw bdm ;191 user defined interrupt vector 136
dw bdm ;192 user defined interrupt vector 137
dw bdm ;193 user defined interrupt vector 138
dw bdm ;194 user defined interrupt vector 139
dw bdm ;195 user defined interrupt vector 140
dw bdm ;196 user defined interrupt vector 141
dw bdm ;197 user defined interrupt vector 142
dw bdm ;198 user defined interrupt vector 143
dw bdm ;199 user defined interrupt vector 144
dw bdm ;200 user defined interrupt vector 145
dw bdm ;201 user defined interrupt vector 146
dw bdm ;202 user defined interrupt vector 147
dw bdm ;203 user defined interrupt vector 148
dw bdm ;204 user defined interrupt vector 149
dw bdm ;205 user defined interrupt vector 150
dw bdm ;206 user defined interrupt vector 151
dw bdm ;207 user defined interrupt vector 152
dw bdm ;208 user defined interrupt vector 153
dw bdm ;209 user defined interrupt vector 154
dw bdm ;210 user defined interrupt vector 155
dw bdm ;211 user defined interrupt vector 156
dw bdm ;212 user defined interrupt vector 157
dw bdm ;213 user defined interrupt vector 158
dw bdm ;214 user defined interrupt vector 159
dw bdm ;215 user defined interrupt vector 160
dw bdm ;216 user defined interrupt vector 161
dw bdm ;217 user defined interrupt vector 162
dw bdm ;218 user defined interrupt vector 163
dw bdm ;219 user defined interrupt vector 164
dw bdm ;220 user defined interrupt vector 165
dw bdm ;221 user defined interrupt vector 166
dw bdm ;222 user defined interrupt vector 167
dw bdm ;223 user defined interrupt vector 168
dw bdm ;224 user defined interrupt vector 169
dw bdm ;225 user defined interrupt vector 170
dw bdm ;226 user defined interrupt vector 171
dw bdm ;227 user defined interrupt vector 172
dw bdm ;228 user defined interrupt vector 173
**initvec: org $0000** ; put the following reset vector information ; at address $00000 of the memory map
dw $00f0 ; zk=0, sk=f, pk=0
dw $0200 ; pc=200 -- initial program counter
dw $f3fe ; sp=f3fe -- initial stack pointer
dw $0000 ; iz=0 -- direct page pointer

**org $000c**

**swi_vec: dw oclint ;**

**org $0080**

**tof_vec: dw tofint ; jump vector for tof interrupt handler**

**icl_vec: dw iclint ; jump vector for ic1 interrupt handler**

* This is the end of the org00008.asm file.

* Title : InitVec.ASM

* Description : This file is included to set up the reset vector ($00000 - $00006), and the GPT module

* IC1 interrupt vector ($0082,$0083)
ic2_vec: dw ic2int  ; jump vector for ic2 interrupt handler
ic3_vec: dw ic3int  ; jump vector for ic3 interrupt handler
oc1_vec: dw oc1int  ; jump vector for oc1 interrupt handler
oc2_vec: dw oc2int  ; no interrupt for oc2
oc3_vec: dw oc3int  ; no interrupt for oc3
oc4_vec: dw oc4int  ; no interrupt for oc4
ic4_oc5_vec: dw bdm  ; no interrupt for ic4/oc4
tof_dvec: dw bdm  ; no interrupt for tof here since it
                      ; has been promoted
paovf_vec: dw bdm  ; no interrupt for pulse accumulator overflow
paif_vec: dw bdm  ; no interrupt for pulse accumulator input flag
pit_vec: dw pitint  ; jump vector for periodic interrupt handler

***********************************************************************
* Title: initsys
* Description: initialize & configure system including
*               the software watchdog and system clock.
***********************************************************************

initsys:  ; give initial values for extension registers
           ; and initialize system clock and cop

   ldab  #$0f
   tbek  ; point ek to bank f for register access
   tbyk  ; point yk to bank f
   ldab  #$00
   tbxk  ; point xk to bank 0
   ldab  #$01
   tbzk  ; point zk to bank 1

   ; ldd  #$00cf  ; make imb available to cpu
   ; std  simmcr  ; (for some reason this statement crashes the debugger)
   bclr  sypcr,$80  ; turn cop (software watchdog) off,
                      ; since cop is on after reset
   ldaa  #$7f
   staa  syncr  ; set system clock to 16.78 mhz
   ldd  #$0003  ; at reset, the csboot block size is 512k.
   std  csbarbt  ; this line sets the block size to 64k since
                      ; that is what physically comes with the evb16
   ldd  #$0103
   std  csbar0  ; set up 64k ram in page 1 of address space
   std  csbar1
   std  csbar2
   ldd  #$fff8
   std  csbar3  ; set up 2k block for external duart
                  ; (lcd uses first 2 bytes of this block)
   ldd  #$7830  ; no wait states for the pseudo-rom
   std  csorbt
   ldd  #$5030
std csor0
; cs0 -> upper byte, write only
ldd #$3030
std csor1
; cs1 -> lower byte, write only
ldd #$7830
std csor2
; cs2 -> both bytes, read and write
; ldd #$38f0
; std csor3
; cs3 -> 3 wait states, read and write
ldd #$3b70
std csor3
; cs3 -> lower byte, read and write, 13 wait states

ldd #$02ff
; csboot chip select for 16-bit port
; evb board also requires cs0,cs1 & cs2
; to be configured as chip selects
; cs3 is chip select for lcd (8-bit port)
; addr23/cs10 is set for eclk output

std cspar0

ldd #$0000
std portc
; make the pins on portc that will be
; configured as output ports low
ldd #$0000
; pc1,pc2,pc3,pc4,pc5,pc6,pc7 configured
; as output ports instead of chip selects

std cspar1
bclr pepar,#fff
; configureporte as i/o port
bset porte0,#fff
bset ddre,#ef
; configure porte as i/o output port
; except pe4 (the negate phase bit)

bclr pfpar,#fff
; configure portf as i/o port
bclr portf,#fff
; make the pins on portf that will be
; configured as outputs low
bclr ddf,#fff
; configure portf as input port
ldd #$064c
; periodic interrupt request level 6
std picr
; periodic interrupt vector is $0c
ldd #$0002
; periodic timer clock not prescaled (122 us per count)
std pitr
; periodic timer modulus is 2 (i.e. 4.098 khz)

*****************************************************************************
* Title : initram
* Description : initialize the hcl6's 1k internal sram
* (put sram in memory map at $fff000 to $fff3ff)
* and set the stack inside it.
*****************************************************************************

initram:
;initialize internal sram and stack
ldd #$00ff
std rambah
; store high ram array, benk f
ldd #$f000
std rambal
; store low ram array
clr rammcr
; enable ram
; set sk to bank f for system stack
lds  #$f3fe  ; put sp at top of 1k internal sram
ldx  #$f000  ; initialize ram contents...
ldy  #$f000
ramloop:
ldd  0,x
std  0,y
aix  #2
aix  #2
cpx  #$f3fe
bne  ramloop

/**************************************************************************
* Title: InitADC.ASM
* Description: Initializes the A/D module for 10-bit conversions
* with a 9 uSecond conversion time.
**************************************************************************

InitADC:
LDD  #$0000    ; enable A/D module
STD  ADCMCR
LDD  #$0083    ; configure A/D for 10-bit, 9 uSec conversions
STD  ADCTL0    ; write to ADCTL1 to start a conversion

/**************************************************************************
* Title: InitQSM.ASM
* Description: Initializes the Queued Serial Module for operation
* with the MAX537 12-bit serial DAC.
**************************************************************************

InitQSM:
*  ldaa  #$80    ; qsm disabled
*  staa  qmcr    ; don't set iarb
ldaa  #$0f ;
staa  qilr    ; disable interrupts
ldaa  #$08
staa  qpdr    ; sck inactive low, pcs0 inactive high
ldaa  #$0a
staa  qpar    ; assign pcs0 & mosi pins to qsp module
ldaa  #$ff
staa  qddr    ; configure qspi pins as outputs
ldd  #$8002
std  spcr0    ; master, cpol=cpha=0, 4.19 mhz, 16-bit transfer
ldd  #$0000
std  spcr2    ; newqp=0, endqp=0; 1 word sent

; fill command ram:
ldaa  #$40
staa cr0 ; cont=0, bitse=1, pcs2,1,0 = 000, assert cs0
; don't use spcr1 delay to sck, no delay between
; (16 bit transfer to dac1)
staa cr1
staa cr2
staa cr3 ; only 4 command registers may be used

; fill transmit ram (addresses and zeroes):
1dd  #$1000
std  boostdac ; load dac1 with zeros
1dd  #$5000 ;
staa buckdac ; load dac2 with zeros
1dd  #$9000 ;
staa tr2 ; load dac3 with zeros
1dd  #$d000 ;
staa tr3 ; load dac4 with zeros
boostdac equ  tr0  ; -> boost current control dac (12 bits)
buckdac equ  tr1  ; -> phase current control dac (12 bits)
torquedac equ  tr2  ; -> set torque echo dac (12 bits)
brakedac equ  tr3  ; -> set brake echo dac (12 bits)

*****************************************************************************
* Title: InitGPT.ASM
* Description: configure input capture and output compares
* for phase locking hall effect sensor signals.
* and interrupt processing
*
*****************************************************************************
InitGPT:
1daa  #$00
staa  gptpdr ; write low to gpt output port
1daa  #$78 ;
staa  gptpddr ; make output compare pins outputs

1dd  #$0081
std  gptmcr ; iarb=1 for gpt module
1dd  #$9240
std  gpticr ; int. request level =2, int. vector base =40h
; (icl vector is 41h, or address 82 & 83h)
; ocl is promoted to highest priority
; (ocl vector address is 80 & 81)

input_capture:
; input capture channel 4 function enabled.

    ldaa  pactl
    anda  #$fb
    oraa  #$04
    staa  pactl
; input capture channel 1 both edges.
; input capture channel 2 both edges.
; input capture channel 3 both edges.
; input capture channel 4 is disable.
   ldaa #$3f
   staa tctl2
; input capture channel 1 interrupt enable.
; input capture channel 2 interrupt enable.
; input capture channel 3 interrupt enable.
; input capture channel 4 interrupt disable.
   ldaa tmsk1
   anda #$78
   oraa #$07
   staa tmsk1
; system clock divide by 256.
; timer overflow interrupt is enabled.
   ldaa tmsk2
   anda #$f8
   oraa #$86
   staa tmsk2
output_compare:
; output compare 5 function enabled.
   ldaa pactl1
   anda #$fb
   staa pactl1
; oc2..oc4 connected to timer logic
   ldaa #$3f
   staa tctl1
; all output compare interrupts are disabled
   ldaa tmsk1
   anda #$07
   oraa #$70
   staa tmsk1
; reset oclm7.
   ldaa #$00
   staa oclm
; all force compare bits are off.
   ldaa cforc
   anda #$07
   staa cforc
; timer output compare register 5 is set to $ffff
   ldd #$ffff
   std ti405
; timer output compare resister 1 is set to $0f
   ldd #$0f
   std toc1
pwm:
; disable discrete output from fla bit, driven out on the pwma pin.
; disable discrete output from flb bit, driven out on the pwmb pin.
   ldaa cforc
   anda #$fc
   staa cforc
; system clock divide by 4
; normal pwm operation on pwma.
; the higher speed of pwma is selected
; normal pwm operation on pwmb.
; the higher speed of pwmb is selected
  ldaa #$10
  staa pwmc
; duty cycle for channel a is set to 5%
  ldaa #$0d
  staa pwma
; duty cycle for channel b is set to 5%
  ldaa #$0d
  staa pwmb
pam_gated:
; pulse accumulator overflow interrupts disabled.
; pulse accumulator interrupts disabled.
  ldaa tmsk2
  anda #$af
  staa tmsk2
; gated time accumulation mode is set.
; pulse accumulator is enabled.
; a zero on pai inhibits counting.
; system clock divided by 512.
  ldaa pactl
  anda #$04
  oraa #$60
  staa pactl
; pulse accumulator counter is set to $00
  ldaa #$00
  staa pacnt

*******************************************************************************
* File Name : MACROS.ASM
* Description : This file contains all macros used in this thesis
* Note : This program is written for the M68HC16Z1EVB.
*******************************************************************************

$macro write_boostdac ; transmit serial data to boostdac (takes
  ldd #$0000 ; about 4 us for 1 transmission)
  std spcr2 ;
  ldd #$8000 ; start transmission
  std spcr1 ;
$macroend
$macro write_buckdac ; transmit serial data to buckdac (takes about
  ldd #$0101 ; 4 us for 1 transmission)
  std spcr2 ;
  ldd #$8000 ; start transmission
  std spcr1 ;
$macroend
$macro write_torquedac ; transmit serial data to torquedac (takes
1dd  #0202  ; endqp=2, newqp=2
std  spcr2
1dd  #8000  ; start transmission
std  spcr1
$
macro write_brakedac
1dd  #0303  ; endqp=3, newqp=3
std  spcr2
1dd  #8000  ; start transmission
std  spcr1
$
macro rupdx
bcc rdx  ; if no carry, just exchange result
aix #01  ; increment the quotient by one
rdx:  xgdx  ; put quotient in d
$
macro ton_hea
bset gptpdr,#heap1 ; take oc2 pin high
psha
1daa tctl1
bclr tctl1,#03  ; disconnect oc_pin from timer
staa tctl1
pula
$
macro toff_hea
bclr gptpdr,#heap1 ; take oc2 pin low
psha
1daa tctl1
bclr tctl1,#03  ; disconnect oc_pin from timer
staa tctl1
pula
$
macro ton_heb
bset gptpdr,#hebp1 ; take oc3 pin high
psha
1daa tctl1
bclr tctl1,#0c  ; disconnect oc_pin from timer
staa tctl1
pula
$
macro toff_heb
bclr gptpdr,#hebp1 ; take oc3 pin low
psha
1daa tctl1
bclr tctl1,#0c  ; disconnect oc_pin from timer
staa tctl1
pula
$

270
$macro ton_hec  ; take oc4 pin high
    bset gtpdr,#hecpl;
psha
    ldaa tctl1
    bclr tctl1,#$30 ; disconnect oc_pin from timer
    staa tctl1
    pula
$macroend

$macro toff_hec
    bclr gtpdr,#hecpl;
psha
    ldaa tctl1
    bclr tctl1,#$30 ; disconnect oc_pin from timer
    staa tctl1
    pula
$macroend

$macro ton_buck_led
    bset porte0,#buckled;
$macroend

$macro toff_buck_led
    bclr porte0,#buckled ;
$macroend

$macro ton_boost_led
    bset porte0,#boostled ;
$macroend

$macro toff_boost_led
    bclr porte0,#boostled
$macroend

$macro set_init_pin
    bset porte0,#init
$macroend

$macro clr_init_pin
    bclr porte0,#init
$macroend

$macro rdbuf_and_address
    ldd #instr_reg        ; result in accum a, bit7 is busy flag
                ; the rest is contents of address counter
$macroend

$macro pushab
    psha
    pshb
$macroend

$macro pushba
    pshb
    psha
$macroend

$macro pullab
    pulb
    pula
$macroend

$macro pullba
pula
pulb

$macroend
toff_gate_drives ; disable motor drive
  bset portc,#$04;
$macroend
ton_gate_drives ; engage motor drive
  bclr portc,#$04;
$macroend
lock_motor ; lock motor on phase a-b
  bset portc,#$08;
$macroend
unlock_motor ; unlock motor
  bclr portc,#$08;
$macroend
forward_rotation ; motor should rotate in the forward direction
  bset portc,#$10;
$macroend
reverse_rotation ; motor should rotate in the reverse direction
  bclr portc,#$10;
$macroend
motor_mode ; motor - as opposed to regenerate
  bset portc,#$20;
$macroend
regen_mode ; regeneration mode
  bclr portc,#$20;
$macroend
buck_mode ; chop battery voltage
  bset portc,#$40;
$macroend
boost_mode ; boost battery voltage
  bclr portc,#$40;
$macroend
curr_mode_cont ; implement current mode control
  ldd #$0707;
  std pwmregs;
  bset portc,#$02;
$macroend
duty_cycle_cont ; implement duty-cycle control
  bclr portc,#$02;
$macroend

*****************************************************************************
* Title : InitMOTOR.ASM
* Description : Initialize MOTOR DRIVE AND CONTROL SETTINGS
* *****************************************************************************
InitMOTOR: ; Initialize MOTOR DRIVE AND CONTROL SETTINGS
  idaa gptpdr ; load the state of the hall effect sensors
  anda #$07 ;
asla ; shift left until bit one becomes bit 5
asl
asl
asl ; bit 1 is now bit 5
staa gptpdr
ldaa tctl1
bclr tctl1,#$3f ; disconnect oc2,oc3&oc4 from timer output
; and output what is in gptpdr
staa tctl1
bclr flags,#fault

initmotl:
clear comment ; clear the torque ripple compensation pointer
toff_gate_drives
forward_rotation
duty_cycle_cont
motor_mode
unlock_motor
buck_mode

*****************************************************************************
* Title: INITKEYPAD
* Description: Initialize the keypad and store mode settings in
*               FLAGS
*****************************************************************************
INITKEYPAD: ;* Initialize Keypad
set_init_pin
brn initkeypad
brn initkeypad
clr_init_pin
jsr read_keypad ; store initial keypad setting in flags

*****************************************************************************
* Title: InitLCD.ASM
* Description: Initialize the LCD Display
*****************************************************************************
InitLCD: ; Initialize LCD
ldd #$ffff
ilcd0: subd #$0001
bne ilcd0
lda #38
staa instr_reg ; function set (8 bit interface)
ldd #$fff
ilcd1: subd #$0001
bne ilcd1
lda #38h
staa instr_reg
ldd #$fff
ilcd2: subd #$0001
bne ilcd2
lda #38h
staa instr_reg ; repeat function set (8 bit interface)
jsr wait_busy_flag ;
lda #38h ;
staa instr_reg ; 2 lines, 5x7 font
jsr wait_busy_flag ;
lda #08h ;
staa instr_reg ; turn display off
jsr wait_busy_flag ;
 lda #0eh ;
staa instr_reg ; display on, cursor on, don’t blink
jsr wait_busy_flag ;
 lda #01h ;
staa instr_reg ; clear display and return the cursor home

**********************************************************************************************************************************************
* File Name : DISPLAY.ASM
*
* Description : This is the main program.
* Initializes peripherals
* Reads buttons and execute button commands
* Process Hall Effect (Input Capture) and
* control (Periodic Timer) interrupts.
* Note : This program is written for the M68HC16Z1EVB.
**********************************************************************************************************************************************

$include "..\equates.asm" ; table of equates for register addresses
$include "..\init\org00008.asm" ; point all unused interrupts to bdm
$include "..\init\vec.asm" ; initialize reset and interrupt vectors
    org $0200 ; start program after interrupt vectors
**** initialization routines *****
$include "..\init\sys.asm" ; initially set ek=f, xk=0, yk=f, zk=1
    ; set sys clock at 16.78 mhz, disable cop
$include "..\init\ram.asm" ; initialize and turn on sram
    ; set stack (sk=f, sp=f3fe)
$include "..\init\adc.asm" ; configure a/d for 10-bit, 9 usec conversion
$include "..\init\qsm.asm" ; configure qspi to interface with
    ; power electronics module and display
$include "..\init\gpt.asm" ; configure icl interrupt
**** include macros *****
$include "..\macros\macros.asm" ; lcd macros and other stuff
$include "..\init\motor.asm" ; motor drive and control settings
$include "..\init\keypad.asm" ; initialize keypad
**** lcd initialization routine *****
$include "..\init\lcd.asm" ; initialize lcd
**** start of user program area *****
$set left_justified ; if this conditional assembly variable is
    ; set, all numbers on lcd display are
    ; left-justified. if not, the numbers will
    ; be right-justified, i.e. numbers on lcd may
    ; have leading spaces

274
jsr clear_display
jsr cursor_off
ton_gate_drives
duty_cycle_cont
buck_mode
jsr offset_null ; go null all analog current channels
ldd #$8000
tdp ; allow interrupts
start:
jsr carriage_return
ldaa #2
staa dec_places
ldd fil_brake_cmdh
lde #$50 ; max pot location is 50% phase advance
emul
ldx #$3ff
ediv
xgdx
; brake pot controls phase advance
std percent_adv

ldd temp_reg ;
lde #$1000
emul
ldx #$2002
ediv
xgdx
jsr disnumu
jsr message0
db ' v ' 
ldaa #2
staa dec_places
ldd fil_brake_cmdh
lde #$1000
emul
ldx #$2002
ediv
xgdx
jsr disnumu
jsr message0
db ' v ' 

jsr goto_line2
ldd fil_torque_cmdh
lde #$450
ldx #max_const_torque
emuls
edivs
rupdx
clr dec_places
jsr disnums
jsr message0
db ' nm \'
ldd new_torque_cmd
lde #1450
ldx #max_const_torque
emul
ediv
rupdx
clr dec_places
jsr disnums
jsr message0
db ' nm \'
jsr goto_line3
ldd dc_bus_volth
lde #1400 ; 100 v --> 1ff
ldx #$7fe
emul
ediv
rupdx
clr dec_places
jsr disnums
jsr message0
db ' v \'
ldd dc_boost_volth
lde #1400
ldx #$7fe ; 100 v --> 1ff
emul
ediv
rupdx
clr dec_places
jsr disnums
jsr message0
db ' v \\
jsr goto_line4
testb: ldd dc_phase_currh
lde #1100
ldx #$1ff ; 100 a --> 1ff
emuls
edivs
xgdx
clr dec_places
jsr disnums
jsr message0
db ' \\
lde #1100 ; 100 a --> 1ff
ldx #$1ff ; current multiplied by 2 after a/d
emuls
edivs
xgdx
clr dec_places
jsr disnums
jsr message0
db ' \
ldd motor_rpm
clr dec_places
jsr disnumu
jsr message0
db ' rpm   \
jsr set_drive_mode
brclr flags,#fault,mlcont
jsr clear_display
jsr message0
db ' !! fault!!  \
jmp bdm
mlcont: jmp start

iclint:  pshm d,e,x ; save registers
brset gptpdr,#$01,icl_high; test if icl is set since
;we interrupt on both edges
jmph icl_low;

icl_high: ton_hea
ldaa to_count ; determine if timer has overflowed twice
clr to_count ; reset the timer overflow counter
cmpa #$01 ;
blt icl_tok ; timer is ok
beq icl_ckof ; check overflow

icl_ovf: ton_hea
bset flags,#tof ; set the timer overflow flag for
; use by the other two hall effect channels
ldd #$ffff ;
std new_hea_period ;
movw ticl,old_tic1 ; store the rising edges
jmp icl_chk_comm ; go do commutation stuff

icl_ckof: ldd tic1 ; check if counter has overrun
subd old_tic1 ;
bcc icl_ovf ; counter overrun

icl_tok: ldd tic1 ;
subd old_tic1 ;
std new_hea_period ;
movw ticl,old_tic1 ; store the rising edges
bclr flags,#tof ;
ldd motor_rpm ;
cpd #min_pav_rpm ; minimum rpm above which we do
; phase advance
bgt  ic1_phase_adv ; do phase advance
ldd  hea_periodh ; period of hall effect a from pll
jmp  ic1_pav_skip ; skip phase advance

ic1_phase_adv:
    ldd  hea_periodh ;
    jsr  calc_pav_deltat ; calculate phase advance delta t

ic1_pav_skip:
    addd tic1 ; adjust tic1 with the amount of phase advance
    std  toc2 ; load toc2 for output compare
    jmp  ic1_chk_comm ;

ic1_low:
    toff_heah ; take hea pin low

ic1_chk_comm:
    lde  tic1 ;
    jsr  torque_shp_sync ; go sync the torque shaping interrupts

reset_ic1: bclr tf1g1,#$01 ;
pulm d,e,x ; restore accumulator d,e,x
rti ; return from interrupt

ic2int: pshm d,e,x ; save registers
brset gptpdr,#$02,ic2_high ; test if ic2 is set since
                            ; we interrupt on both edges
    jmp  ic2_low ;

ic2_high: ton_heb
    brset flags,#tof,ic2_ovf ; if timer overflow flag set
                            ; then turn on the output compare pin
    jmp  ic2_tok ; timer ok

ic2_ovf: ton_heb ; timer overflow, or speed below
          ; min_oc_rpm so turn on the oc pin
    ldd  #$ffff ;
    std  new_heb_period ;
    movw tic2,old_tic2 ;
    jmp  ic2_chk_comm ; go do commutation stuff

ic2_tok:
    ldd  tic2 ;
    subd old_tic2 ;
    std  new_heb_period ;
    movw tic2,old_tic2 ;
    ldd  motor_rpm ;
    cpd  #min_pav_rpm ; minimum rpm above which we do
                     ; phase advance
    bgt  ic2_phase_adv ; do phase advance
    ldd  heb_periodh ; period of hall effect b from pll
    jmp  ic2_pav_skip ; skip phase advance

ic2_phase_adv:
    ldd  heb_periodh ;
    jsr  calc_pav_deltat ; calculate phase advance delta t

ic2_pav_skip:
    addd tic2 ; adjust tic2 with the amount of phase advance
std toc3 ; load toc3 for output compare
jmp ic2_chk_comm ;
ic2_low: toff_heb ; take heb pin low
ic2_chk_comm:
  lde tic2 ;
  ;
  jsr torque_shp_sync ; go sync the torque shaping interrupts
reset_ic2: bclr tf1gl1,#$02 ;
pulm d,e,x ; restore accumulator d,e&x
rti ; return from interrupt
ic3int: pshm d,e,x ; save registers
  brset gptpdr,#$04,ic3_high ; test if ic3 is set since
  ; we interrupt on both edges
  jmp ic3_low ;
ic3_high:ton_hec
  brset flags,#tof,ic3_ovf ; if timer overflow flag set
  ; then turn on the output compare pin
  jmp ic3_tok ; timer ok
ic3_ovf: ton_hec ; timer overflow, or speed below
  ; min_oc_rpm so turn on the oc pin
  ldd #$ffff ;
  std ::ew_hec_period ;
movw tic3,old_tic3 ;
jmp ic3_chk_comm ; go do commutation stuff
ic3_tok: ldd tic3 ;
  subd old_tic3 ;
  std new_hec_period ;
movw tic3,old_tic3 ;
  ldd motor_rpm ;
cpd #min_pav_rpm ; minimum rpm above which we do phase
  ; advance
  bgt ic3_phase_adv ; do phase advance
  ldd hec_periodh ; period of hall effect c from pll
  jmp ic3_pav_skip ; skip phase advance
ic3_phase_adv:
  ldd hec_periodh ;
  jsr calc_pav_deltat ; calculate phase advance delta t
ic3_pav_skip:
  addd tic3 ; adjust tic3 with the amount of phase advance
  std toc4 ; load toc4 for output compare
  jmp ic3_chk_comm ;
ic3_low: toff_hec ; take heb pin low
ic3_chk_comm:
  lde tic3 ; tic3
  ;
  jsr torque_shp_sync ; go sync the torque shaping interrupts
reset_ic3: bclr tf1gl1,#$04 ;

279
pulm d,e,x ; restore accumulator d,e&x
rti ; return from interrupt

oclint: rti

pshm d,e,x,y;
brset gptpdr,#$08,clear
bset gptpdr,#$08
jmp comm1

clear: bclr gptpdr,#$08

comm1: clre ;
ldd heabc_period ;
ldx #({maxsteps}*!6) ;
ediv ; interrupt #maxsteps times within a
; commutation interval
rupdx ; round up result and exchange d and ix
tst commcnt ;
beq new_comm ;
addd tocl ; add the time interval till next interrupt
jmp sto_next ;

new_comm: addd commtime ; sync to hall effect sensors
bset tmstk,#$08 ; enable ocl interrupts for torque shaping

sto_next: std tocl ; store in tocl for next output compare
cira
ldab commcnt ;
aslb ; two bytes per entry
tde
ldx #trcomp_tab ; modify command torque
ldd e,x ; get the percent fraction torque ripple boost
lde fil_torque_cmdh ; get the commanded torque
asle
fmuls ;
adce fil_torque_cmdh ;
ste comp_torque_cmd ; this is the compensated torque command
; for this interval

:* diagnostic
ldy #buckdac
jsr setup_dac
write_buckdac

lda commcnt ; advance the current profile counter
inca
cmpa #maxsteps ; check if we have exceeded the limit
bls sto_cnt ; store the count
movw fil_torque_cmdh,comp_torque_cmd ; if we ever exceed
; this, then don’t do
; any torque shaping
lda #maxsteps ; if so then fix it at maximum
sto_cnt: staa commcnt ;
ocl_reset: bclr tflgl,#$08 ;
pulm d,e,x,y ; restore registers
rti ;

c2int:
;     pshm d
;     ldd hea_periodh
;     addd old_tic1
;     std toc2
;     bclr tflgl,#$10 ;
;     pulm d
rti

c3int:
;     pshm d
;     ldd heb_periodh
;     addd old_tic2
;     std toc3
;     bclr tflgl,#$20 ;
;     pulm d
rti

c4int:
;     pshm d
;     ldd hec_periodh
;     addd old_tic3
;     std toc4
;     bclr tflgl,#$40 ;
;     pulm d
rti

tofint: pshm d ;
ldaa to_count ;
inca
staa to_count
cmpa #$02 ; permissible maximum count
blt tof_reset ; timer has not overflowed more than
; once so exit
ldaa #$02
staa to_count
bset flags,#tof ;
tof_reset: bclr tflg2,#$80 ; clear the timer overflow flag
pulm d ;
rti ; return from interrupt

;*******************************
;* modify this routine to sequentially null all
;* three channels. store a lookup table of commutation
;* pattern in code space with pointer to the appropriate
;* offsets, and use the appropriate offsets in computation
;* whenever needed
;* disable all interrupts when entering this routine
;*****************************************************************

offset_null: pshm d,e,x,y,z ; save registers
clr
std  pwmda
std  pwnregs ; turn off phases
jsr  clear_display
jsr  message0
db  ' calibration \'
jsr  goto_line2
jsr  message0
db  ' in progress ...
jsr  goto_line3
jsr  message0
db  '.. one moment .. \'
jsr  goto_line4
jsr  message0
db  'nulling phase a\'
lda  portc ; save the state of portc pins (these
          ; have been set by initmotor)
psha
lda  tctl1 ; save the oc control settings
psha
lda  gptpdr ; save the hall effect settings
psha
forward_rotation
clr  pwma
clr  pwmb ; make pwm channels quiet
bclr  tctl1,#$3f ; disconnect output compare pins from timer
             ; check:
        bset  gptpdr,#heap1 ; take oc2 high
        bclr  gptpdr,#hebp1 ; take oc3 low
        bset  gptpdr,#hecp1 ; take oc4 high , 101 selects phase a (no
                        ; negation)
        ldy  #iphasea_zerop ; where phase a (non-inverted) offset
                        ; should be stored
        brset  porte0,#$10,null_error ; neg_phase signal should be clear
jsr  null_channel ; go compute channel a offset
        bclr  gptpdr,#heap1;
        bset  gptpdr,#hebp1;
        bclr  gptpdr,#hecp1 ; 010 selects phase a (negated)
        ldy  #iphasea_zerop; where the phase a (inverted) offset
                        ; should be stored
        brclr  porte0,#$10,null_error ; neg_phase signal should be high
jsr  null_channel ; go compute channel a inverted offset
jsr  goto_line4
jsr  message0
db  'nulling phase b\'
bset  gptpdr,#heap1 ; take oc2 high
bset  gptpdr,#hebp1 ; take oc3 high
bclr  gptpdr,#hecp1 ; take oc4 low , 110 selects phase b
ldy  #iphaseb_zerop ; where phase b offset should be stored
brset porte0,$10,null_error ; neg_phase signal should be clear
jsr null_channel ; go compute channel b offset
bclr gptpdr,#heapl ; take oc2 low
bclr gptpdr,#hebpl ; take oc3 low
bset gptpdr,#hecpl ; take oc4 high , 001 selects phase b
                   ; inverted
ldy #iphaseb_zeron ; where phase b offset should be stored
brclr porte0,$10,null_error ; neg_phase signal should be set
jsr null_channel ; go compute channel b offset
jsr goto_line4 ;
jsr message0
db 'nulling phase c'
bset gptpdr,#heapl ; take oc2 high
bclr gptpdr,#hebpl ; take oc3 low
bclr gptpdr,#hecpl ; take oc4 low , 100 selects phase c
                   ; (negated)
ldy #iphasec_zeron ; where phase c negated offset should be stored
brclr porte0,$10,null_error ; neg_phase signal should be set
jsr null_channel ; go compute channel c offset
bclr gptpdr,#heapl ; take oc2 high
bset gptpdr,#hebpl ; take oc3 low
bset gptpdr,#hecpl ; take oc4 low , 011 selects phase c
ldy #iphasec_zerop ; where phase c offset should be stored
brset porte0,$10,null_error ; neg_phase signal should be clear
jsr null_channel ; go compute channel c offset
pula ; the hall effect input settings
staa gptpdr ;
pula ; the timer control settings
bclr tctl1,#$3f ; disconnect oc2, oc3 & oc4 from timer
                   ; output
staa tctl1 ;
pula
staa portc
jsr clear_display ;
pulm d,e,x,y,z ; restore the regisiters
rts

null_error:
   jsr clear_display
   jsr message0
db ' !! error !! \'
jsr goto_line2
jsr message0
db ' in commutation\'
jsr goto_line3
jsr message0
db ' pal \'
jmp bdm

null_channel:
pshm y ; save the result pointer
ldz  #$0fff  ; output should settle $fff of times
osn0:  ldd  #$0030  ;
std  adctl1  ; scan 4 channels (an0 - an3) but we are
          ; only interested in phase current
          ; and boost inductor current
          brclr  adstatccf,#$01,osn1  ; wait for data
          ldd  rjurr0  ; the phase current
          std  new_phase_curr;
          lde  #$7d9  ; filter cutoff frequency parameter
          ldy  #dc_phase_currh;
          jsr  lowpass_filter
osn1:  brclr  adstatccf,#$02,osn2  ; wait for data
          ldd  rjurr1  ; the boost inductor current
          std  new_boost_curr;
          lde  #$7d9  ; filter cutoff frequency parameter
          ldy  #dc_boost_currh;
          jsr  lowpass_filter;
          ldd  dc_phase_currh;
          cpd  new_phase_curr;
          bne  osn0;
          ldd  dc_boost_currh;
          cpd  new_boost_curr;
          bne  osn0;
          aiz  #$-1;
          bne  osn0;
          ldd  dc_phase_currh  ; the result
pull  y  ; the destination pointer
std  0,y  ; store the result
movw  dc_boost_currh,ilb_zero;
rts

pitint:  pshm  d,e,x,y,z

atod:  ldd  #$0030
std  adctl11  ; single conversion on all 8 channels
          ; do some computations before the the
          ; atod conversion is done (10 us)
ldab  gptpdr  ; get the position of the motor from
          ; hall effect signals
andb  #$70  ; save only the 0c4..0c2 bits
stab  temp1  ; temporary store it
asrb
asrb
asrb
          ; shift only 3 times because we have
          ; 2 bytes per entry
cmpb  #$00
beq  bad_atod  ; multiplexer error (bail out for now)
cmpb  #$0e
beq  bad_atod  ; multiplexer error (bail out for now)
ldaa  gptpdr
anda  #$70  ; save only the oc4..oc2 bits
cmpa templ
bne bad_atod ; commutation occurred so bail out
bset flags,#good_atod ; a/d value is good
cla
brclr portc,#forw,atod1 ;
orb #$10 ; if forw bit is set then set the
; msnibble (remember x2)
atod1:
tde
ldx #pho_adj_tab ; phase offset adjust table
ldd e,x ; get the pointer to the offset
xgdy ; the pointer should point to data space
wait_atod: brclr adstatccf,$01,wait_atod ; if phase current
; conversion not done, then wait
; bset flags,#good_atod,atod2 ; if value not good then don't
; store
subd 0,y ; subtract the offset
std new_phase_curr ; this is the new phase current
atod2: bset flags,#good_atod ; set flag for the case of two bad a/d
; in a row
brset portc,inh,no_loop ; if for any reason we have
; disengaged the drive disengage control loop
ldd dc_boost_volth ;
bleq no_input ; if bus voltage is zero then do nothing
jmp buck_loop
bad_atod: brclr flags,#good_atod,atod2 ; exit a/d conversion
; if we’ve been here before
bclr flags,#good_atod ; set a flag to indicate one
; unsuccessful a/d
jmp atod ; go try one more conversion
no_loop:
clrw buckm_interrl
clrw buckm_interrh
clrw boostm_interrl
clrw boostm_interrh
toff_buck_led
toff_boost_led
jmp phase_lock

buck_loop: buck_mode
    duty_cycle_cont ;
movw pwmdata,pwmregs ; write the pwm data from previous comput.
new_buck_calc:
    ldd dc_boost_volth ; check for bus overvoltage
    lde bemf
    adde #$3ff
    cpe #$ff8 ; 800 V for now
    ble maxv_ok
    lde #$ff8
maxv_ok: ste max_voltage ;
cpd #$ff8 ; 800 V for now
max_voltage
; turn off boost converter
add
; get the back emf voltage
add #1ff
; 100 V over the back emf
xgcd
; desired boost voltage in X
lde dc_bus_volth
; the bus voltage
lde #ff
; unity duty cycle
ediv
xgcd
; vin/vout_desired is in D
lde #ff
; e contains f?-(vin*ff)/vout_desired
bcs no_boost
; if vin > vout_desired, then no_boost
beq no_boost
; 0.7 max duty cycle
gle duty_ok
lde #b2
; set to max duty cycle
jmp duty_ok
; zero duty cycle
ton_buck_led
toff_boost_led
jmp duty_ok1
ton_boost_led
toff_buck_led
ted
; store the boost duty cycle
stb pwmdata
new_loop: ldd buckm_interrl
lde buckm_interrh
pshm d,e
; ldd comp_torque_cmd
; does not work unless we have interrupts
lde #ki_buck_mot
; the chopper integrator constant
emuls
; compute ki * error
add #ki_buck_mot
; integrate the control error
adce buckm_interrh
stb buckm_interrh
; store the integrator error
; (number stored it actual
lde #kp_buck_mot
emuls
add #ki_buck_mot
adce buckm_interrh
; output of controller is
; (d*vin-vemf) * 255
add #emi_by_ffl
; add low word of back_emf*ff
adce #emi_by_ffh
; add high byte
jsr limited
; get the bus voltage from the output filter
lde dc_bus_volth
;
edivs ; could round result
bmi tbm_neg_sat ; torque, buck, motor, negative
; saturation (** go do regen)
cpx #$ff ; max duty cycle
bgt tbm_pos_sat ; positive saturation
pulm d,e ; restore the stack
xgdx ; answer in l
std pwndata ; answer --> pwmb, to be loaded
; on next interrupt
jmp phase_lock ;

; tbm_neg_sat:
clrw pwndata ; 0 -> pwmb ; to be loaded on
; next interrupt
pulm d,e
sted buckm_interrh ; don't update the integrator
jmp phase_lock ;

; tbm_pos_sat:
ldd #$ff ;
std pwndata ; ff-> pwmb , to be loaded on next
; interrupt
pulm d,e ; get the pushed integrator state
sted buckm_interrh ; don't update the integrator

; no_input:
jmp phase_lock ;

; speed_loop:
jmp phase_lock ;

phase_lock:
    ldd new_phase_curr
    jsr pil ; phase lock the hall-effect signals

reset_pitr:
    pulm d,e,x,y,z
    rti
mux_error: ton_boost_led
    jmp atod

bdm: ton_buck_led
    jmp bdm
    rti

;******************************************************************************
;* phase current offset adjust table
;******************************************************************************
pho_adj_tab:
;* reverse (i.e. msb = 0)
dw $0000 ; 000 never happens (but keep here to
; keep indexing simple)
dw iphasetb_zerop ; 001 phase b
dw iphasetb_zerop ; 010 phase a
; forward (i.e. msb=1)
dw $0000 ; 000 never happens (but keep here to
; keep indexing simple)
dw iphaseb_zeron ; 001 phase b negated
   ;
dw iphasea_zeron ; 010 phase a negated
   ;
dw iphasec_zeron ; 011 phase c
   ;
dw iphaseb_zero ; 101 phase a negated
   ;
dw iphasec_zeron ; 110 phase b
   ;
dw $0000 ; 111 never happens (but keep here to
   ; keep indexing simple)

;******************************************************************************************
;* torque ripple compensation table. entry is the signed percent *
;* fraction by which torque should be increased *
;******************************************************************************************

trcomp_tab: ; torque ripple compensation table
   dw !0 ;(!90 * $8000)/!100) ; i.e. 90 %
   dw !0 ;(!70 * $8000)/!100) ; i.e. 70 %
   dw !0 ;(!40 * $8000)/!100) ; i.e. 40 %
   dw !0 ;(!20 * $8000)/!100) ; i.e. 20 %
   dw !0 ;(!10 * $8000)/!100) ; i.e. 10 %
   dw !0 ;(!5 * $8000)/!100) ; i.e. 5 %
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0
   dw !0

*   dw {0 - (!!20 * $8000)/!100)} ; i.e. -5 %
*   dw {0 - (!!50 * $8000)/!100)} ; i.e. -10 %
*   dw {0 - (!!80 * $8000)/!100)} ; i.e. -20 %
$include "..\sbrts\subs.asm"

******************************************************************************************
* File Name: SUBS.ASM
* Description: This file contains all subroutines
* Note: This program is written for the M68HC16Z1EVB.

;********************************************************************;
;* this subroutine is used for lcd display operations *
;* it saves accumulator a and waits until lcd busy flag *
;* is cleared. *
;********************************************************************;

wait_busy_flag:
psha ; save accum a on stack
wbf0: brclr instr_reg,#80h,rtn_bf ; if bit 7 clear, then exit loop
   ldaa #$ff ;
wbf1: deca
   bne wbf1 ;
   bra wbf0 ;
 rtn_bf: pula ; pop a from the stack
           rts ;

;********************************************************************;
;* this subroutine displays in-line messages written *
;* in code space on the lcd display. end of message *
;* is indicated by the \ character. values in *
;* registers ix,iy,d,e are clobbered, along with values *
;* in temp1, and temp2 if you need any of these, then *
;* save them before entering this subroutine. *
;********************************************************************;

message0: pullba ; pop the condition code
tde ; put condition code in register e
pullba ; pop the return pc address
subd #0002h ; popped address is to bytes higher
xgde ; exchange d and e, cc in d, pc-2 in e
std temp1 ; store cc in temporary register 1
ste temp2 ; store pc-2 in temporary register 2
xgex ; pc-2 in ix
1de #0000h ;
message1: ldaa e,x ; get first character
adde #01h ; increment pointer
cmpa #\" ; check if terminator
beq message2 ; if yes, skip down
jsr write_character ; takes an ascii in accum a
bra message1 ; loop for next character
message2: adde #02h ; add the two we subtracted
adde temp2 ; add the pc-2
xgde ; put the new pc in ab
pushba ; put it on the stack
ldd templ ; get the condition codes
pushba ; put it on the stack
rts

;********************************************************************
;* this subroutine returns the cursor home. the contents *
;* display is cleared of data display ram remains unchanged *
;********************************************************************
cursor_home:
  jsr    wait_busy_flag ; wait until busy flag is clear
  ldaa   #02h          ; return cursor home
  staa   instr_reg    ; does not erase contents of ddram
  rts

;********************************************************************
;* this subroutine clears the display and returns the cursor home. *
;* display is cleared of data display ram remains unchanged *
;********************************************************************
clear_display:
  jsr    wait_busy_flag ; wait until busy flag is clear
  ldaa   #01h          ; clear the display
  staa   instr_reg    ;
  rts

;********************************************************************
;* this subroutine shifts the cursor one character to the left      *
;********************************************************************
shift_cursor_left:
  psha                        ; push data to be written on stack
  jsr    wait_busy_flag ; wait until busy flag is clear
  ldaa   #10h           ; shift cursor to the left
  staa   instr_reg    ;
pula            ; restore data to be written
  rts

;********************************************************************
;* this subroutine turns the cursor on and blinks the cursor:       *
;********************************************************************
cursor_on_blink:
  jsr    wait_busy_flag ; wait until busy flag is clear
  ldaa   #0fh           ; turn on cursor and blink
  staa   instr_reg    ;
  rts

;********************************************************************
;* this subroutine turns the cursor on but does not blink the cursor *
;*****************************************************************************

cursor_on_noblink:
    jsr    wait_busy_flag ; wait until busy flag is clear
    ldaa   #0eh          ; turn on cursor and blink
    staa   instr_reg    ;
    rts

;*****************************************************************************
;* this subroutine turns the cursor off *
;*****************************************************************************

cursor_off:
    jsr    wait_busy_flag ; wait until busy flag is clear
    ldaa   #0ch          ; turn on cursor off
    staa   instr_reg    ;
    rts

;*****************************************************************************
;* this subroutine shifts the display to the left *
;*****************************************************************************

shift_display_left:
    jsr    wait_busy_flag ; wait until busy flag is clear
    ldaa   #18h          ; shift display to the left
    staa   instr_reg    
    rts

;*****************************************************************************
;* this subroutine sets the character generator ram address *
;* 6-bit address is assumed to be in accumulator a *
;*****************************************************************************

set_cgram_address:
    jsr    wait_busy_flag ; wait until busy flag is clear
    anda   #3fh          ; clobber 2 msbs
    oraa   #40h          ; set the function code: 01xxxxxx
    staa   instr_reg    ; set cgram address to xxxxxx
    rts

;*****************************************************************************
;* this subroutine sets the data display ram address *
;* 6-bit address is assumed to be in accumulator a *
;*****************************************************************************

set_ddram_address:
    jsr    wait_busy_flag ; wait until busy flag is clear
    anda   #7fh          ; clobber 2 msbs
    oraa   #80h          ; set the function code: 1xxxxxxx
    staa   instr_reg    ; set ddram address to xxxxxx

291
rts

;******************************************************************************
;* this subroutine positions the cursor at the beginning of  
;* the first line.                                *
;******************************************************************************
carriage_return:   ; position cursor at beginning of 1st line
    jsr   wait_busy_flag ; wait until busy flag is clear
    ldaa  #00h         ; takes the 6-bit address
    anda  #7fh         ; clobber 2 msbs
    oraa #80h         ; set the function code: 1xxxxxxx
    staa instr_reg    ; set ddram address to xxxxxxx
    rts

;******************************************************************************
;* this subroutine positions the cursor at the beginning of  
;* the second line.                                      *
;******************************************************************************
goto_line2:
    jsr   wait_busy_flag ; wait until busy flag is clear
    ldaa  #40h         ; takes the 6-bit address
    anda  #7fh         ; clobber 2 msbs
    oraa #80h         ; set the function code: 1xxxxxxx
    staa instr_reg    ; set ddram address to xxxxxxx
    rts

;******************************************************************************
;* this subroutine positions the cursor at the beginning of  
;* the third line.                                      *
;******************************************************************************
goto_line3:
    jsr   wait_busy_flag ; wait until busy flag is clear
    ldaa  #10h         ; takes the 6-bit address
    anda  #7fh         ; clobber 2 msbs
    oraa #80h         ; set the function code: 1xxxxxxx
    staa instr_reg    ; set ddram address to xxxxxxx
    rts

;******************************************************************************
;* this subroutine positions the cursor at the beginning of  
;* the fourth line.                                      *
;******************************************************************************
goto_line4:
    jsr   wait_busy_flag ; wait until busy flag is clear
    ldaa  #50h         ; takes the 6-bit address
    anda  #7fh         ; clobber 2 msbs
    oraa #80h         ; set the function code: 1xxxxxxx
staa instr_reg ; set ddr address to xxxxx
rts

;******************************************************************************
;* this subroutine writes an ascii character in accumulator *
;* a to the current cursor position *
;******************************************************************************

write_character:
    jsr wait_busy_flag ; wait until busy flag is clear
    staa data_reg ; write data in data register
wcl: rts

;******************************************************************************
;* this subroutine reads what ascii character is written at *
;* the current cursor position. result is in accumulator a *
;******************************************************************************

read_character:
    jsr wait_busy_flag ; wait until busy flag is clear
    ldaa data_reg ; read ascii data, data is in accum a
    rts ;

******************************************************************************
* this subroutine displays the keypad settings mode on the lcd *
* display *
******************************************************************************

disp_mode: jsr carriage_return ; carriage return
            brset flags,$80,torque_mode ; if torque bit set
            ; goto torque_mode
            jsr message0 ; write "speed" message
db ' [m] speed \'
            ;
bra disp_mode2 ; skip next two lines
            torque_mode: jsr message0 ; write "torque" message
db ' [m] torque \'
            ;
disp_mode2: jsr goto_line2 ; position cursor on line 2
            brset flags,$40,forw_mode ; if forward bit set
            ; goto forw_mode
            jsr message0 ; write "reverse" message
db ' [o] reverse \'
            ;
bra disp_mode3 ; skip next two lines
            forw_mode: jsr message0 ;
            db ' [o] forward \'; write "forward" message
            disp_mode3: jsr goto_line3 ; position cursor on line 3
            brset flags,$20,drive_mode ; if drive bit set
            ; goto drive_mode
            jsr message0 ; write "coast" message
db ' [d] coast \'
bra disp_mode4 ; skip next two lines
            drive_mode: jsr message0 ;
            db ' [d] drive \'; write "drive" message
disp_mode4:  jsr goto_line4  ; position cursor on line 4
  brset flags,#$1e, start_mode ; if start_bit set
    ; goto start_mode
  jsr message0  ; write "stop" message
  db ' [e] stop \' ;
  bra dm_exit  ; skip next two lines
start_mode:  jsr message0  ; write "start" message
  db ' [e] start \' ;
dm_exit:     rts  ; return from subroutine

;******************************************************************************
;* this subroutine delays the for a period indicated by contents *
;* of accumulator d                                                 *
;******************************************************************************

delay:     lda #000fh  ;
loop0:     jsr read_keypad  ; read keypad while you wait
  brset flags,#mode_chg,dl_exit; if mode changes, the exit
  sube $0001h  ;
  bne loop0  ;
  subd $0001h  ;
  bne delay  ;
dl_exit:    rts

;******************************************************************************
;* this subroutine displays which control loop is currently active *
;******************************************************************************

loop_msg:   jsr carriage_return  ;
  brset flags,#$80,tlmsg1
  jsr message0
  db ' [* speed loop *] \'
  bra lpmsg1
tlmsg1:     jsr message0
  db ' [ torque loop ] \'
lpmsg1:     jsr goto_line2
  jsr message0
  db ' [*** active ***] \'
  jsr goto_line3
  jsr message0
  db 'torque: 450 nm \'
  jsr goto_line4
  jsr message0
  db 'speed : 1100 rpm \'
  rts

;******************************************************************************
;* this subroutine displays inverter’s input and ouput dc voltage *
;* and current.                                                     *
;******************************************************************************
vi_msg:    jsr carriage_return
          jsr message0
          db 'v_in = 255 v  \'
          jsr goto_line2
          jsr message0
          db 'i_in = 39.6 a  \'
          jsr goto_line3
          jsr message0
          db 'v_out = 877 v  \'
          jsr goto_line4
          jsr message0
          db 'i_out = 14.8 a  \'
          rts

;**********************************************************************
;* this subroutine displays the inverters input and ouput powers         *
;* and the boost converter's efficiency                                *
;**********************************************************************

pow_msg:    jsr carriage_return
          jsr message0
          db ' inverter power  \'
          jsr goto_line2
          jsr message0
          db 'p_in = 12.7 kw \'
          jsr goto_line3
          jsr message0
          db 'p_out = 11.5 kw \'
          jsr goto_line4
          jsr message0
          db 'bc_eff = 94.5 %  \'
          rts

;**********************************************************************
;* this subroutine reads the keypad settings and updates the mode       *
;* status in the upper nibble of flags                                *
;**********************************************************************

read_keypad:
    pushab ; push accumulator ab on stack
    ldaa portf ; read keypad
    anda #$f0 ; only upper nibble is used
    ldab flags ;
    andb #$f0 ; get rid of the other status bits
    cba ; see if there is change
    beq rkexit ; it is the same so quit
    ldab flags ; get the flags again
    andb #$0f ; clobber upper nibble
    aba ; add b to a
    staa flags ; store keypad in upper nibble of flags

295
bset flags,#mode_chg ; set the mode change flag
rkexit: pullab       ; restore accumulator ab
        rts

;***************************************************************************
;* this subroutine to display 2's complement number in accumulator d *
;* with sign, leading zero blanking, and decimal point. number        *
;* of digits after decimal point is stored in memory location         *
;* dec_places. allow for a total of six characters                    *
;***************************************************************************
disnums:
tstd                ; test whether number is negative
bpl disnums0         ; if positive number, skip down
negd                ; if negative, make it a positive number
tde                 ; put the number in accumulator e
        ldaa '#-'     ; initial sign
        bra disnums1  ; skip down
disnums0: tde        ; put the number in accumulator e
        ldaa '#'      ; put a space
disnums1: ldaab #$6   ; 6 digits total
        subb dec_places ;
         stab dec_places ;
         staa leading_sign ;
        ted            ; put number to be displayed back in d
        bset flags,$#04 ; set the leading zero flag
        ldx #$10000    ; display ten thousands digit
        jsr disnums2   ;
        ldx #$1000    ; display one thousands digit
        jsr disnums2   ;
        ldx #$100     ; display one hundreds digit
        jsr disnums2   ;
        ldx #$10      ; display tens digit
        jsr disnums2   ;
        ldx #$1       ; display units digit
        jsr disnums2   ;
        rts            ; return
disnums2: pushab      ; save the number
        ldaab dec_places ;
        decb
        stab dec_places ; store back the decimal place counter
        bne disnums3    ; if not zero, skip down
        ldaa #'.'      ; if zero, display decimal point
        bclr flags,$#04 ; clear leading sign flag
        jsr write_character
        jsr disnums2
        pullab         ; restore the number
        ; leading sign still in a
        idiv           ; quotient in ix, remainder in d
        aix #'0'       ; add the ascii zero offset
        xgdx           ; remainder in ix, ascii number in b
        brclr flags,$#04,disnums7 ; if leading zero flag is
; cleared, skip down
cmpb #0
bne disnums5 ; if no, skip down
ldaa dec_places
cmpa $1 ; check decimal point after next digit
beq disnums5 ; if yes, skip down

$if left_justified
xgdx ; {assemble for left justified number
   ; display}
rts ; {assemble for left justified number
   ; displays}
$endif

ldab ' ' ; if no, init for space
bra disnums7 ; display it
disnums5: ldaa leading_sign ;

$if left_justified
cmpa ' ' ; {assemble for left justified number displays}
beq disnums6 ; {assemble for left justified number displays}
$endif

jsr write_character;
disnums6: bclr flags,#$04 ; clear leading zero flag
disnums7: tba
   jsr flags,#$04
   xgdx
   rts

;**********************************************************************************************
; * this displays an unsigned binary number in accumulator d * 
; * with leading zero blanking, and decimal point. number * 
; * of digits after decimal point is stored in memory location * 
; * dec_places. allow for a total of six characters * 
;**********************************************************************************************

disnumu: tde
   ldaa ' ' ; put a space
   ldab #$6 ; 6 digits total
   subb dec_places ;
   stab dec_places ;
   staa leading_sign ;
ted ; put number to be displayed back in d
bset flags,#$04 ; set the leading zero flag
ldx #100000 ; display ten thousands digit
jsr disnumu0000 ;
ldx #1000 ; display one thousands digit
jsr disnumu2
ldx #100 ; display one hundreds digit
jsr disnumu2
ldx #10 ; display tens digit
jsr disnumu2
ldx #1 ; display units digit

jsr  disnumu2 ;
rts    ; return

disnumu2: pushab   ; save the number
ldab dec_places ;
decb
stab dec_places  ; store back the decimal place counter
bne  disnumu3   ; if not zero, skip down
lda a #.'       ; if zero, display decimal point
bclr flags,#$04 ; clear leading sign flag
jsr  write_character ;
disnumu3: pullab   ; restore the number
            ; leading sign still in a
idiv     ; quotient in ix, remainder in d
aix #0'   ; add the ascii zero offset
xdgdx    ; remainder in ix, ascii number in b
brclr flags,#$04,disnumu7 ; if leading zero flag is
            ; cleared, skip down
cmpb #0'   ;
bne disnumu5 ; if no, skip down
ldab dec_places
lda a disnumu7
cmpa #$1    ; check decimal point after next digit
beq  disnumu5 ; if yes, skip down

$if  left_justified
xdgdx       ; {assemble for left justified number displays}
rts          ; {assemble for left justified number displays}
$endif

ldab # '    ; if no, init for space
bra  disnumu7 ; display it
disnumu5: ldaa leading_sign  ;

$if  left_justified
cmpa # '     ; {assemble for left justified number displays}
beq  disnumu6 ; {assemble for left justified number displays}
$endif

jsr  write_character ;
disnumu6: bclr flags,#$04   ; clear leading zero flag
disnumu7: tba 
jsr  write_character;
xdgdx
rts

limite:          ; correct over/under flow of adde instruction
  bvc  eok      ; if no over/under flow, jump to return
  bpl  eunder   ; v=1 and n=0 => underflow
  lde  #$7fff   ; v=1 and n=1 => overflow, set e at maximum
  rts          ; return from subroutine
  eunder: lde  #$8000 ; set e at minimum
  eok: rts      ; return from subroutine

limitd:          ; correct over/under flow of addd instruction
bvc   dok ; if no over/under flow, jump to return
bpl   dunder ; v=1 and n=0 => underflow
ldd   #$7fff ; v=1 and n=1 => overflow, set d at maximum
rts   ; return from subroutine
dunder: ldd   #$8000 ; set d at minimum
dok:   rts   ; return from subroutine

limited: ; correct over/under flow of addd/e instructions
bvc   edok ; if no over/under flow, jump to return
bpl   edunder ; v=1 and n=0 => underflow
lded   #$7fff ; v=1 and n=1 => overflow, set ed at maximum...
ldd   #$ffff
rts   ; return from subroutine
edunder: lded   #$8000 ; set ed at minimum...
ldd   #$0000
edok:   rts   ; return from subroutine
tedwsx: ; transfer e to d with sign extend
ted   ; move e contents to d
bmi   negsgn ; see if lsword is negative
cldre ; if lsword is not negative, extend plus
rts   ; return from subroutine
negsgn: lded   #$ffff ; if lsword is negative, extend minus
rts   ; return from subroutine

;********************************************************************

;* this subroutine reads the keypad and set: the appropriate *
;* drive mode, filters all a/d data and computes motor speed *
;********************************************************************

set_drive_mode:
  jsr   calc_motor_rpm ; compute the motor rpm
  ldd   motor_rpm
check_out_comp:
  cpd   #min_oc_rpm   ; below this speed we don’t do output compares
  bge   set_direction ; check keypad settings
  ldd   tcnt
  addd   #$ffff ; advance output compare timer with maximum count
  std   toc2
  std   toc3
  std   toc4
set_direction:
  bset portf,#forw,forward_dir ;
reverse_rotation ; motor reverse
  jmp   chk_fault
forward_dir:
  forward_rotation ; motor forward
chk_fault:
  jmp   no_fault ; diagnostic **
ldaa  portf

299
anda #50e ; keep the fault bits
beq no_fault ;
bset flags,#fault
gates_off:
   toff_gate_drives ;
jmp filter_signals ; fault condition so don’t bother to check drive
no_fault:
   bclr flags,#fault ;
brCLR portf,#driv,gates_off ; if coast then turn gates off
brCLR portf,#strtr,gates_off ; if stop then turn gates off for now
   ldd winding_temp ;
cpd #max_temp ;
; bge gates_off ;
   ldd endturn_temp ;
cpd #max_temp ;
; bge gates_off ;
drive:
   unlock_motor ;
ton_gate_drives ;
filter_signals: brCLR adstatccl,ilb_f,filter_vboost ; go filter vboost
filter_iphase:
   ldd new_phase_curr ; get the selected phase current.
   ; appropriately negated in control routine
   lde #ki_dc ; the filter parameter
   ldy #dc_phase_currh ; pointer to high byte of where the
   ; output is stored
   jsr lowpass_filter ; go find dc component
filter_ilboost:
   ldd rjurr1 ; get the a/d value of the boost current
   subd ilb_zero ; subtract the offset
   asld ; multiply by 2 so that 100a -> 1ff
   std new_boost_curr ;
   lde #ki_dc ;
   ldy #dc_boost_currh ; pointer to high byte of filtered boost
   ; current
   jsr lowpass_filter ; go find dc component
filter_vboost: brCLR adstatccl,#vbst_f,filter_vbus
   ldd rjurr2 ; get the a/d value of the boost voltage
   lde #$6 ;
   emul ; multiply by 6 to make 100v -> 1ff
   std new_boost_volt ; store the result
   lde #ki_dc ; load filter constant
   ldy #dc_boost_volth ; load pointer to high byte of output
   jsr lowpass_filter ; go find dc component
   ldd dc_boost_volth ;
   ldd new_boost_volt
   lde #$ff ;
   emul
   std vboost_by_ffh ; store vboost*ff for control
filter_vbus: brCLR adstatccl,#vbus_f,filter_torque
   ldd rjurr3 ; get the a/d value of the boost voltage

300
asld             ; multiply by 2 to make 400v --> 7fe
std   new_bus_volt ; store the result
lde   #ki_dc      ; load filter constant
ldy   #dc_bus_volth; load pointer to high byte of output
jsr   lowpass_filter; go find dc component
ldd   dc_bus_volth;
    lde   #$ff
    emul
    std   vbus_by_ffh ; store bus voltage * ff for control

filter_torque: brcrlr adstatccf,#torq_f,filtbrke_torque
    ldd   rjurr4      ; get the a/d value of the torque command
    std   temp_reg;
    lde   #$1ff        ; scale potentiometer reading to 1ff (max)
    emul
    ldx   #$3fc        ; $3fc instead of 3ff to allow some
                        ; headroom for pot
    ediv
rupdxx
    pshm  d             ; potential new torque command is pushed
                        ; on stack
                        ; pot has been scaled to max
                        ; torque. 100a --> 3ff
    ldd   motor_rpm;
    cpd   #corner_rpm   ;
    bgt   tspd e
    lde   #max_const_torque; below base speed, so limit commanded
                        ; torque to maximum constant torque
    ste   max_perm_torque; maximum permissible torque
    jmp   ft2 e

    tspd:
    lde   #max_trpm_hi  ;
    ldd   #max_trpm_lo  ;
    ldx   motor_rpm    ;
    ediv
rupdxx
    std   max_perm_torque; maximum constant power permissible torque
    ble   ft3 e

    ft2:
    pulm  d             ; the potential new torque command
    cpd   max_perm_torque; compare potential torque to
                        ; max permissible torque
    bls   ft3 e          ; torque not exceeded so potential
                        ; torque is fine
    ldd   max_perm_torque; clamp at maximum permissible torque

    ft3:
    std   new_torque_cmd;
    lde   #ki_dc        ;
    ldy   #fil_torque_cmdh;
    jsr   lowpass_filter;

filter_brake: brcrlr adstatccf,#brk_f,filtbrke_wind_temp
    ldd   rjurr5        ;
std new_brake_cmd;
lde #ki_dc
ldy #fil_brake_cmdh;
jsr lowpass_filter;

filter_wind_temp: brclr adstatccf,#wtemp_f,filter_endt_temp;
ldd rjurr6;
std winding_temp;

filter_endt_temp: brclr adstatccf,#etemp_f, sdm_rtn

ldd rjurr7;
std endturn_temp;

; now compute boost multiplication factor

ldd dc_boost_volth
subd dc_bus_volth
bpl boostok ; output volts < input volts
clrw mult_fac ; zero multiplication factor
rts

boostok:
lde #!5 ; 2*244e-6s/100uh
emul;
std mult_fac ; this number is used by boost controller

sdm_rtn: rts

;******************************************************************************

;* data to be filtered in d. filter ki in e. pointer to where
;* data is stored in iy

lowpass_filter:

subd 0,y ; xh[n]-yh[n-1]
emuls ; error * ki
addd 2,y ; add in yl[n-1]
adce 0,y ; add in yh[n-1]
ste 0,y ; store high byte
std 2,y ; store low byte
rts

calc_motor_rpm:

brclr flags,#tof,cmrpm

ldd #$ffff;

std hea_periodh
std heb_periodh
std hec_periodh
std new_hea_period
std new_heb_period
std new_hec_period

zero_rpm:

clrd

std bemf
std bemf_by_ffh
std  bemf_by_ffh
std  motor_rpm
rts

cmrpm: clre ; calculate motor speed
    ldd  hea_periodh 
    addh  heb_periodh 
    adce #0 
    addh  hec_periodh 
    adce #0 
    ldx  #$03
    ediv
    bcc  cmrpm1 
    aix  #$01

cmrpm1:
    stx  heabc_period
    ldx  Rpm_count/$10000 
    ldd  Rpm_count%$10000
    ediv
    rupdx 
    cpd  #!7
    bhs  good_rpm 
    jmp  zero_rpm

good_rpm:
    std  motor_rpm 
    lde  #k_emf ; back emf per rpm constant
    emul
    ldx  #!100 ; bemf constant had a factor of 100 in it
    ediv
    xgcdx ; answer in d
    std  bemf
    lde  #$ff
    emul 
    sted  bemf_by_ffh ; back emf * ff
    rts

;******************************************************************************
;* this subroutine loads up data in accumulator e into the *                *
;* dac pointed to by iy.  *                                            *
;******************************************************************************

setup_dac:
    cpe  #$07ff  ; maximum positive number
    bgt  sdacl
    cpe  #$f800  ; minimum negative value
    blt  sdacl2
    sdacl:  ldd  #$0800  ; the 12-bit dac offset
             ade  
             ldd  0,y 
             get the current dac address & data
             andd  #$f000  ; save the address bits
             ade  ;
ste  0,y ; put the new data & address back
rts
sdacl: lde  #$07ff ; upper limit
  jmp  sdac3 ;
sdac2: lde  #$f800 ; lower limit
  jmp  sdac3 ;

;****************************************************************************
;* this subroutine calculates the time between the input *
;* capture and rising edge under phase advance *
;* the hex_period is passed in d
;****************************************************************************
calc_pav_deltat:
  lde  percent_adv ;
  nege ;
  adde  #$100 ;
  emul ;
  ldx  #$100 ;
  ediv ;
  rupdx ; round up result and exchange d and ix
  rts

;****************************************************************************
;* this subroutine updates commtime with ticx. it disables *
;* ocl interrupts if speed is too high or timer has *
;* overflowed. it calls a software interrupt to sync ocl *
;* interrupts to motor commutation. ticx is passed in e *
;****************************************************************************
torque_shp_sync:
  brset flags,#tof,doc1 ; if timer has overflown,
                          ; disable torque shaping
  ldd  motor_rpm ;
  cpd  #max_shp_rpm ;
  bgt  doc1 ;
  clr  commcnt ; we commutate on each edge
  ste  commtime ; ticx --> commtime
  swi
  rts ;
doc1:  bclr  tmsk1,#$08 ; disable ocl interrupts
         movw  fil_torque_cmdh,comp_torque_cmd ; no torque shaping
  rts ;

;****************************************************************************
;* this subroutine phase locks the period of the hall-effect *
;* sensors *
;****************************************************************************
p11:  ldy  #new_hea_period ;
jsr pl_loop;
ldy #new_hec_period;
jsr pl_loop;
ldy #new_hec_period;
jsr pl_loop;
rts

pl_loop: ldd 0,y ; new hall effect period
cpd 2,y ; hex_periodh
bhi pll_pos_err
beq pll_zero_error;

pll_neg_err:
  ldd 2,y ; hex_periodh
  subd 0,y ;
  lde #ki_pll;
  emul ;
  pshm e
  lde 4,y ; hex_periodl
  sde ;
  ste 4,y ;
  pulm d ;
  lde 2,y ;
sbce #0
  sde
  ste 2,y ;
  bcc pll_rtn
; clrre
; clrd
  jmp pll_rtn

pll_zero_error:
  rts

pll_pos_err:
  subd 2,y ; hex_periodh
  lde #ki_pll
  emul
  addd 4,y ; hex_periodl
  adce 2,y ; hex_periodh
;  bcc pll_rtn ;
;  ldd #$ffff ;
;  tde
 pll_rtn: ste 2,y ; hex_periodh
 std 4,y ; hex_periodl
rts
References


[18] Private conversation with Prof. C. C. Chan. For more information, contact: The General Manager (Mr. Zhang Xin Yu), Tsinghua University, Yin-na High Tech Company, P. O. Box 1021, Beijing 102201. Phone: 86 10 255-3731, ext. 279 or 86 10 977-1350, ext. 331. Fax: 86 10 977-1874.


[45] *Nonoriented sheet steel for magnetic applications*, United States Steel International Inc., 600 Grand Street, Pittsburgh, PA 15230, USA.


309


[56] *PC-146B Hi-Flux™ Powder Cores*, The Arnold Engineering Company, Marengo, IL. USA.


[60] *MC68HC16Z1, User's Manual*, Motorola Literature Distribution; Phoenix, AZ.

[61] *MC68HC16Z1, System Integration Module Reference Manual*, Motorola Literature Distribution; Phoenix, AZ.


[63] *MC68HC16Z1, CPU16 Reference Manual*, Motorola Literature Distribution; Phoenix, AZ.

[64] *MC68HC16Z1, Queued Serial Module Reference Manual*, Motorola Literature Distribution; Phoenix, AZ.

[65] *MC68HC16Z1, General Purpose Timer Reference Manual*, Motorola Literature Distribution; Phoenix, AZ.