Wirelessly Enabled Control of Cyber-Physical Infrastructure with Applications to Hydronic Systems

by

Michael Bruce Kane

**A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Civil Engineering)
in the University of Michigan
2014**

Doctoral Committee:

Associate Professor Jerome P. Lynch, Chair
Assistant Professor Lars Junghans
Professor Victor C. Li
Assistant Professor Jeffrey T. Scruggs

DEDICATION

To all of those
in the past, present, and future
that cede some of their life
to learn more about the world
and share that wealth with others

ACKNOWLEDGEMENTS

It would be impossible to show my gratitude for what has been given to me by others in the few short words of this section. However, I will do my best.

The work presented herein was financially supported by the U.S. Navy Office of Naval Research under contracts N00014-05-1-0596 and N00014-09-C0103, the National Science Foundation (NSF) through grant numbers CCF-0910765 and CMMI-0846256, US Department of Commerce National Institute of Standards and Technology (NIST) Technology Innovation Program (TIP) under Cooperative Agreement 70NANB9H908, and the University of Michigan Rackham Graduate School Rackham Graduate Student Research Grant.

Professor Jerome Lynch must first and foremost be acknowledged for his mentorship as research adviser over my last five years at the University of Michigan. Under his direction, the Laboratory for Intelligent Systems and Technologies (LIST) provided me with an environment to pursue new ideas that I felt could impact the world. Prof. Lynch provided encouragement, gentle direction, and the resources I needed to succeed. I would also like to provide my heartfelt gratitude to the rest of my Ph.D. advising committee: Prof. Jeff Scruggs, Prof. Victor Li, and Prof. Lars Junghans. I am lucky that Prof. Scruggs joined the Michigan community mid-way through my studies, so that I could learn from his expertise in civil engineering control systems. Prof. Li is an exemplar civil engineer to look up to, and Prof. Junghans brought a fresh perspective from architecture to my studies in civil engineering. Additionally, I would like to thank Prof. Jason McCormick and Prof. Branko Kerkez for their mentoring and inspiration. Also, my faculty advisers during my studies at

Drexel University, Prof. Emin Aktan, Prof. Frank Moon, and Prof. James Mitchell, should receive thanks, for without their encouragement, a PhD would never have appeared to be an option.

Ann Arbor and the University are blessed with incredible people who have help me along the way. In the LIST group, fellow alumni and post-docs Andy Zimmerman, Andrew Swartz, Junhee Kim, Masahiro Kurata, Yang Liu, Yunhwa Hong, Courtney Peckins, and fellow graduate students Sean O'Connor, Yilan Zhang, Devki Desai, Ted Byrne, Sukhoon Pyo, Jeff Bergman, Mitsuhito Hirose, Nephi Johnson, and Andrew Burton for their friendship, fellowship, and technical assistance. Additionally LIST group undergrads and master's students Zeshi Zheng, Alex Mead, Han Sun, Leon Paredes, and Colin Nangle provided valuable support. Design, fabrication, and laboratory assistance was provided multiple times by Robert Gordenker, Rober Fischer, and Jan Pantolin. Many of the goals in this thesis were achieved due to the motivation provided by my Graduate Roundtable to Increase Profession, Personal, and Educational Development (GRIPPED) peers. I have also had the pleasure of working with the RVTR architecture studio's Geoffrey Thun, Mary O'Malley, Lisa Sauve, and Eric Meyer on fanciful and practical controls projects.

Friends new and old: Ashley Dunfee, the LIST group cast and crew, Tara Clancy, Lingli He, Sherri Cook, Eric Meyer, Mary O'Malley, Lisa Sauve, Matthew Fadden, Ben Cohen, Jeff Dowgala, and numerous others have provided distraction and focus, been through high times and low, and shared stress and freedom.

Outside of Michigan, my graduate career was influenced greatly by my work at the National Center for Research on Earthquake Engineering (NCREE) under Prof. C. H. Loh, P. Y. Lin, Kung Chun Lu, Shu-Hsien Chao, and Shieh-Kung Huang. Experiences abroad in Tokyo, Japan at the 2010 Asia-Pacific Summer School on Smart Structures Technology and in Trento, Italy at the 2012

International Summer School on Smart Structures opened my eyes to international problems and approaches in civil engineering.

The development of the *Martlet* wireless node would not have been possible without the feedback, debugging, and later collaboration of Prof. Andrew Swartz at Michigan Technological University with student Benjamin Winter, and Prof. Yang Wang at Georgia Institute of Technology with students Dapeng Zhu, Xinjun Dong, and Xiaohua Yi.

The field validation of the *Martlet* would not have been possible without Chuck Farrar, Curtt Ammerman, Stuart Taylor, and Eric Flynn at the Los Alamos National Laboratory (LANL); their gracious support was invaluable. I am also grateful to my international colleague Moritz Haeckell from Leibniz Universitat (Hannover, Germany) for his help during the testing of the LANL wind turbine.

The cooling network research was greatly aided by Anthony Seman (Office of Naval Research), Frank Ferrese, David Kocsik, and Dr. Qing Dong (Naval Surface Warfare Center Carderock Division) whom I thank for their technical feedback and support.

Finally, I owe my greatest thanks to my family who have supported me in any and all ways possible during this great journey. My parents Dave and Cathy have been a steadfast presence in my life, being there to listen, advise, and offer support during calls back home which were often spread too far apart.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

ABC	Agent-Based Control
ACK	Acknowledgement
ADC	Analog to Digital Converter
AES	All Electric Ship
ARX	Autoregressive Exogenous
BLEC	Bi-Linear model Efficient Cooling controller (section 4.3.2.4)
BSN	Body Sensor Network
CCA	Clear Channel Assessment
CL	Closed Loop (control)
CLA	Control Law Accelerator
CRC	Cyclic Redundancy Check
COTS	Commercial Off The Shelf
CPS	Cyber-Physical System
CPU	Central Processing Unit
DAC	Digital to Analog Converter
DAQ	Data Acquisition System
DES	Discrete-Event System
EOC	Environmental Operating Conditions
FDD	Frequency Domain Decomposition
FFT	Fast Fourier Transform
GUI	Graphical User Interface
GPIO	General Purpose Input Output
GSL	GNU Scientific Library
HAL	Hardware Abstraction Layer

HDS	Hybrid Dynamical System
HVAC	Heating Ventilation and Air Conditioning
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
ID	Identification
ISR	Interrupt Service Routine
IO	Input-Output
IoT	Internet of Things
LANL	Los Alamos National Laboratory
LEC	Linear model Efficient Cooling controller (section 4.3.2.4)
LMI	Linear Matrix Inequality
LTI	Linear Time-Invariant
LTV	Linear Time-Varying (parameter system)
MCU	Micro-Controller Unit
MIMO	Multiple-Input Multiple-Output
MINLP	Mixed-Integer Non-Linear Program
MLD	Mixed Logical Dynamical (system)
MPC	Model Predictive Control
MR	Magnetorheological
MT	Minimum Time control (section 4.3.2.1)
ODE	Ordinary Differential Equation
OL	Open Loop (control)
OS	Operating System
QL	Quadratic cost with Linear model control (section 4.3.2.2)
QBL	Quadratic cost with Bi-Linear model control (section 4.3.2.2)
PC	Person Computer
PCB	Printed Circuit Board
PID	Proportional-Integral-Derivative (controller)
PSU	Power Supply Unit
PWC	Piece-Wise Constant

PWA	Piece-Wise Affine (system)
PWM	Pulse Width Modulation
RAM	Random Access Memory
RF	Radio Frequency
RPM	Revolutions Per Minute
RTC	Real-Time Clock
RTOS	Real-Time Operating System
SBC	Single Board Computer
SD	Secure Digital (memory card)
SISO	Single-Input Single-Output
SMD	Surface Mount Device
SPI	Serial Peripheral Interface
SPM	Structural Performance Monitoring
STC	Self-Tuning Control
TI	Texas Instruments, Inc.
TPBVP	Two-Point Boundary Value Problem
μSD	Micro-SD (memory card)
UART	Universal (serial) Asynchronous Receiver/Transmitter
UI	User Interface
USB	Universal Serial Bus
VAV	Variable Air Volume
w.r.t.	With Respect To (a variable)
WSN	Wireless Sensor Network

ABSTRACT

Civil infrastructure systems, such as transportation networks, pipe networks, electrical grids, and building environments, are typically managed and controlled with outdated, inefficient, and minimally automated legacy controllers. This is apparent from documented oil pipeline leaks, broad electrical outages, and power plant failures. The relatively recent advents of small inexpensive microcontrollers and low-power wireless networking technologies has revealed opportunities for better managing the operational effectiveness of civil infrastructure systems. Academic research in this field is maturing, yet the field remains in its nascent years of commercial viability, focusing mainly on low data-rate sensing with centralized processing. Little focus has been on distributed wireless control systems for civil infrastructure.

This dissertation follows the development and utilization of a new cyber-physical system (CPS) architecture for civil infrastructure. Embedded computing power is distributed throughout the physical systems and global objectives are met with the aid of wireless information exchange. The *Martlet* wireless controller node was conceived during the first part of this thesis to enable this objective of wirelessly distributed CPS. Once produced, the *Martlet* was used to realize such a controller, motivated by an application in hydronic cooling systems.

The design of the proposed controller began with a study concerning models and objective functions for the control of bilinear systems, like those found in hydronics, when constrained by the resources of a wireless control node. The results showed that previous work with linear quadratic controllers could be improved by using nonlinear models and explicit objective functions.

An agent-based controller utilizing the proposed bilinear model-predictive control algorithm, was then developed accounting for the limitation of, and leveraging the advantages of, wireless control nodes in order to regulate a hydronic system with hybrid dynamics. The resulting *Martlet* based control system was compared to traditional benchmark controllers and shown to achieve adequate performance, with the added benefits of a wireless CPS.

These developments in wirelessly distributed control of complex systems are presented not only with the tested hydronic systems in mind, but with the goal of extending this technology to improve the performance and reliability of a wide variety of controlled cyber-physical civil infrastructure systems.

Chapter 1.

INTRODUCTION

The civil infrastructure that facilitates the function of the global economy is facing problems on a multitude of fronts: these include aging, increasing loads, and new constraints that were not accounted for in the original designs. These intensifying problems necessitate innovative solutions beyond the capabilities of current methods of infrastructure design and operation. Fortunately, technological trends in low-power processing, wireless networking, battery energy density, energy harvesting, and the internet are enabling new approaches. Although civil infrastructure has long had computational and physical components, the recent *cyber-physical systems* (CPS) approach was created to unify the theory of controls, networking, physics, and the interactions thereof in order to leverage enabling technological trends to solve grand challenges (Sha et al. 2008). Embedding computational capabilities deep within the components of a system is an important characteristic of the CPS mindset. Low-power wireless networking has been pivotal to the spread of embedded computing, but most devices currently in use do not meet the need of the latest algorithms for infrastructure control. Such algorithms for nonlinear control themselves need further study and adaption to the objectives and constraints unique to cyber-physical infrastructure systems. This chapter introduces aspects of civil infrastructure that motivate a CPS approach and presents the significant contributions made by this thesis to the advancement of a CPS framework.

1.1 GRAND CHALLENGES FOR INFRASTRUCTURE

In 2012 the National Academy of Engineering listed fourteen grand challenges for engineering in the 21st century. Among them was the need to restore and improve urban infrastructure (The National Academy of Engineering 2012). The roads, bridges, building, pipelines, railways, and power grids that make up the nation's infrastructure are being tested by time, larger loads, and new constraints. It is estimated that the average bridge in the US is 49 years old, many of which were designed for 50 year life-spans. This results in nearly one in four bridges today not meeting the original or current design specification (AASHTO 2008). Similarly, the electrical grid is being taxed with variations in supply from renewable energy and experiencing increases in demand from plug-in electric vehicles (Kempton and Tomić 2005). These vulnerabilities can have catastrophic consequences when not properly managed.

On August 1st, 2007 the 40-year old I-35 bridge over the Mississippi River in Minneapolis, MN collapsed, killing 13 and injuring 121 others. The collapse was partially attributed to a design deficiency that had not been discovered in over 40 years of inspection (NTSB 2007). On July 26th, 2010 an oil pipeline near Marshall, Michigan ruptured, spilling approximately 819,000 gallons of oil and forcing the displacement of over 100 residents due to benzene concerns (Dollhopf and Durno 2011). On March 11th, 2011 a magnitude 9.0 earthquake set in motion a tidal wave that led to meltdowns in three reactors at Japan's Fukushima Dai-ichi power station. Specifically, the chain of events that led to the Fukushima disaster included failures of physical systems, automated control systems, and standard-operating-procedures (Strickland 2011). On August 14th, 2003 a widespread power outage occurred that affected 50 million people across Ohio, Michigan, Pennsylvania, New York, Vermont, Massachusetts, Connecticut, New Jersey, and the Canadian province of Ontario, lasting four days and costing the U.S. economy an estimated $10 billion dollars. The incident was attributed to inadequate understanding of the system, inadequate awareness of system conditions, inadequate execution of existing standard-operating-procedures, and inadequate real-time diagnostic support (Abraham and Efford 2004). These four disasters are only a small subset of numerous recent incidents resulting from failures of infrastructure systems.

No single piece of machinery, electronics, software, or regulation will secure the vulnerabilities in the nation's infrastructure. A multi-pronged approach is required that covers advances in technological hardware, applied controls, systems theory, user interfaces, and even government legislation. Developments will most likely begin in particular application areas, but will need to be abstracted and adapted to address the myriad of problems found throughout the infrastructure engineering field. To this end, this thesis engages these challenges within the realm of hydronic networks, which transfer energy between thermal loads and water. However, the methods presented herein have broad applicability to many problems in infrastructure systems. A CPS framework could ease a number of challenges associated with hydronic systems, structural systems, and infrastructure networks.

1.1.1 Hydronic systems

In many energy systems, thermal energy must be added or removed from an area for the purpose of heating or cooling. The fundamental modes of heat transfer are conduction, convection, radiation, and advection. In hydronic systems, heat is transferred between the object being heated/cooled and water flowing through the object. The water is then typically pumped to another location, transferring the thermal energy along with it. These systems arise in many application areas including building heating and cooling systems, power plant temperature regulation systems, chemical process equipment, and thermoregulation systems (e.g. chilled water systems) onboard naval vessels.

Future naval vessels may be rendered more resilient through the U.S. Navy's explorations of the all-electric ship (AES) design concept. In the AES, an integrated power system combines a ship's two largest plants, the electrical system and powertrain, thereby coupling previously independent engineering plants (Wagner 2007). This thesis considers a chilled water plant and electrical system coupled through the operation of pumps, valves, sensors, and electrical thermal loads (e.g., radar and pulsed weapons systems) (Srivastava et al. 2008). The U.S. Navy is interested in automating these systems to reduce manning requirements on ships without sacrificing fight-through capabilities (Seman III et al. 2003). Effective automated reconfiguration of the interconnected ship plants will require dense arrays of sensors and actuators (Zivi 2002). Building a layer of computational intelligence atop this network will facilitate automated plant reconfiguration in the face of battle damage. As this computational intelligence is pushed down towards sensors and actuators, a rigorous sense-compute-actuate framework, such as those offered by the CPS field, is needed.

1.1.2 Structural systems

Earthquakes, wind, machinery and other dynamic loads can excite a structure to the point where it no longer serves its intended purpose. Structures are traditionally designed from a component point-of-view, in which engineers appropriately size the members and add mass to ensure safety and serviceability under expected loads. The use of passive control systems such as tuned mass dampers (TMD), base isolation, or supplemental damping devices can result in a more economical structure through reduction in mass and/or material (Christopoulos et al. 2006). Proper implementation of semi-active control devices (*e.g.* magnetorheological (MR) dampers, semi-active mass dampers (SMD), and variable stiffness devices (VSD)) has been shown to increase the performance of structural systems beyond improvements possible using passive control (Spencer and Nagarajaiah 2003). However, semi-active control devices in a single structure. A compromise between the complexity of a centralized feedback control system and the performance loss of a decentralized system can be achieved with a distributed control system in which multiple controllers are interconnected. It has been shown that wireless networks of distributed devices can outperform state-of-the-practice decentralized control system when tested experimentally (Linderman 2013; Lynch and Law 2002, 2004; Lynch et al. 2008; Swartz and Lynch 2007, 2009; Wang et al. 2006, 2007, 2009). Distributed networks of wireless sensors and actuators acting as a distributed controllers is, in effect, a CPS. Rigorous CPS frameworks have the potential to improve the performance of semi-active control systems operated using wireless telemetry.

1.1.3 Control of building environments

A more sustainable future for energy efficiency will be attained on three fronts, according to the former United States Secretary of Energy and winner of the Nobel Prize in physics, Steven Chu (Chu 2009). These three fronts are (1) to reduce demand through better efficiency, (2) to increase the supply of renewable energy sources, and (3) for all individuals to do their part on a personal level. The effect of improvements in building energy efficiency could have a tremendous impact on the first of these three challenges. In the year 2008, 40% of the primary energy consumed by the United States was used by the residential and commercial building sector. Of the energy used by the building sector, HVAC systems expended nearly 50% (U.S. Department of Energy 2011). This presents a noteworthy target for efficiency improvements.

HVAC control systems are typically either completely centralized or completely decentralized (Pita 2001). Single family residences and other small buildings have a single zone with a single centralized thermostat which aims to maintain the zone’s temperature within acceptable limits. In larger buildings, spatially varying loads such as solar gains and occupants justify the increased costs of multi-zone systems in which the building is separated into zones consisting of a room or a group of rooms with similar loads. Multi-zone

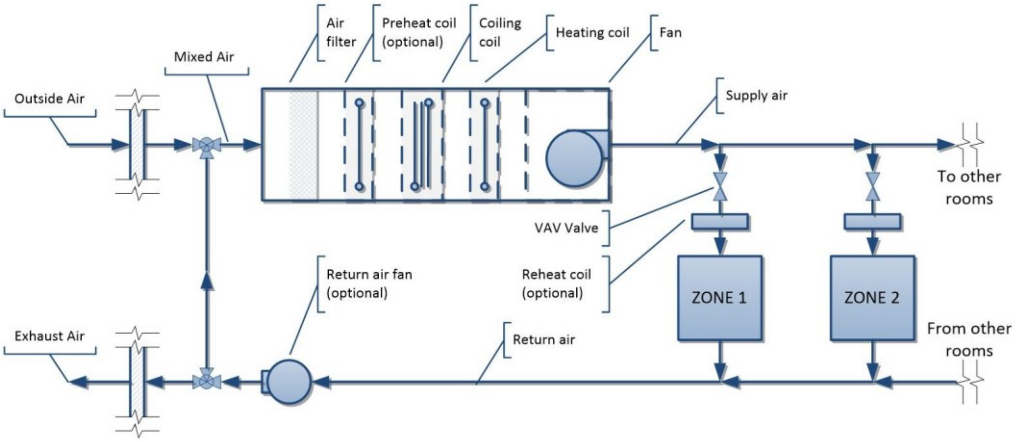


Figure 1.1 General schematic of multi-zone VAV HVAC systems.

variable air volume (VAV) HVAC systems, schematically illustrated in Figure 1.1, can better maintain the desired temperature in each zone. These systems operate through decentralized control of each zone by collocated thermostats and VAV valves. Currently, most VAV systems operate using wired communications and simple distributed controllers that are designed for worst-case thermal and CO_2 loads.

Networks of wireless sensors and their associated computational capabilities can be easily deployed to upgrade these single-zone systems into more efficient multi-zone systems with both local feedback and feedback to the main air handler (Redfern et al. 2006). As wireless sensors with embedded computing are deployed in greater density for controlling building environments more efficiently, a CPS framework could be used to maximize system effectiveness.

1.1.4 Networks and grids

The tremendous breadth of civil engineering covers a many more large-scale, spatially distributed systems utilizing sensing and controls. Canals, levees, and aqueducts have been around for millennia where weirs and locks were manually controlled by skilled personnel. As the systems became more complex, simple distributed programmable logic controllers (PLCs) or supervisory control and data acquisition (SCADA) systems replaced or augmented human feedback and actuation. These distributed computation and control systems are becoming ever more complex in applications areas such as the smart power grid (Strzelecki and Benysek 2008), water management (Fawal et al. n.d.), transportation (Negenborn et al. 2006), and municipal engineering (Kim et al. 1991; Scholze and Zaghloul 2001). However, the computational intelligence, while at a high level in the control architecture, retains only relatively simple PID type controllers at the lowest level. Harnessing the power of today's remarkable embedded computing at the lowest level can increase the systems response time and reliability (Zivi 2002). However, as computing intelligence is migrated away from centralized servers and controllers, and into distributed PLC units, a complex CPS system emerges requiring rigorous system design to ensure system performance, robustness, and resilience.

1.2 ENABLING TRENDS

The power of problem solving algorithms is worthless without the ability to sense and affect the physical world or the ability to compute the solutions in a suitable amount of time. The observation now known as Moore's Law made by Gordon E. Moore in 1965 states that the viable limit of computational power doubles every 18-24 months. This prediction by Moore can be appreciated by observing the infiltration of tremendous computing power into nearly every part of our everyday life (Schaller 1997). Moore's law is now at hold in the embedded processor market resulting in more power-efficient and faster microcontrollers that can be used to operating sensors and actuators, as well as process data generated and used within the system. In effect, computational intelligence is "cheap" to collocate with sensors and actuators.

Similarly, developments in low-power wireless networking have opened up opportunities for sensing and actuating large infrastructure systems that were not viable with legacy wired automation. The paradigm shift from wired to wirelessly networked control systems requires more than a one-to-one replacement of communication links. The centralized wired control system designer limits the number of sensors and actuators in the system in an effort to reduce the computation burden on the centralized controller. Achieving redundancy by duplication of all the wiring and computing further reduces cost effectiveness. Conversely a wireless architecture empowers the control designer with the ability to implement dense arrays of sensors and actuators which would otherwise overwhelm centralized wired control schemes (Zivi 2002). The sharing of the computational load amongst all the agents in the peer-to-peer wireless network maintains computational and communication robustness through redundancy.

The areas showing greatest promise for these technologies are wireless monitoring systems (Johnson et al. 2009; Lynch and Loh 2006; Straser et al. 1998), embedded computing in wireless sensor networks (Logan et al. 2007; Lynch et al. 2004; Nagayama et al. 2009; Swartz et al. 2005; Zimmerman 2007), and wireless feedback control (Graf et al. 2011; HART Communication Foundation 2011; Loh et al. 2007; Lynch et al. 2008; Ploplys et al. 2004; Seman III et al. 2003; Swartz and Lynch 2009; Wang 2009; Wang et al. 2007, 2009). While wireless sensing, wireless control, and in-network computing have all advanced in the past decade, scaling these technologies from a single system to a network or grid of systems remains an open challenge.

1.3 CIVIL INFRASTRUCTURE IN A NEW LIGHT—THE CPS PARADIGM

The term *cyber-physical systems* (CPS) describes a set of interconnecting computational components and physical components that interact in an interdependent way. Similar to *the internet of things* (IoT) concept established a few years prior, the term CPS was coined only eight years ago, and did not see significant usage until the *2006 NSF Workshop on Cyber-Physical Systems* (Zhao et al. 2006). This workshop highlighted the CPS focus on interactions, opposed to the focus of the IoT on ‘things’. Since that time, the CPS research community has continued to grow, bringing the computationally minded cyber-science communities (including control theorists, computer scientists, and networking specialists) together with the physical-science communities (in the electrical, mechanical, aeronautical, biomedical, civil, materials, and social engineering domains). Additionally the systems research community has joined these efforts to address the interfaces and interactions between these heterogeneous systems of systems. A CPS implementation typically requires computational components such as task managers, data storage, encryption services, and software fault diagnostics. It also typically utilizes physical components such as machines, mechanical controllers, and sensors. But, CPS also can include design considerations that are focused on cyber-physical coupling, like bandwidth allocation during increased sensor usage, or security of system vulnerabilities that could lead one to compromise the computational systems in an attempt to adversely affect the physical systems.

This thesis is concerned with the implications of CPS frameworks on civil infrastructure systems, with a specific focus on control systems used to control the performance of a utility-scale system. The study of the physical systems of civil infrastructure has been around for centuries, if not millennia. However, the use of computational components in civil infrastructure system operation is a more recent consequence of the *information age*. These computational components, termed industrial control systems (ICS), can include any combination of the following architectures: supervisory control and data acquisition (SCADA); programmable logic controllers (PLC); and distributed control systems (DCS). Subsystems may also be considered. These include a human machine interface (HMI) providing an operator intuitive access to system’s functions; supervisory computer systems (and backups) for collecting data and sending commands; remote terminal units (RTU) with attached sensor conversion, collection, and transmission modules; PLCs which are more economical and reconfigurable than RTUs; a communication infrastructure; and sensing instrumentation. Typically, these legacy systems put a small amount of computing power (e.g. a PLC) embedded within each of the components of the physical plant, and execute algorithms requiring increased computational effort in a centralized computer system. The new approach of CPS aims to drive the cyber-physical interconnectivity closer to the component levels of the physical plant. This new approach is enabled by wireless technology and low-power computing, creating a new field of wirelessly-enabled CPS, or wireless CPS.

As the CPS community grows, it becomes less focused on each of the cyber- and physical-subsystems, and more on engineering the interconnection of the heterogeneous subsystems during design and operation (Sztipanovits et al. 2012). Examples of these studies in interdependencies include, but are not limited to, system security (Banerjee et al. 2012), validation (Pajic and Mangharam 2012), and unified modeling frameworks (Derler et al. 2012). This thesis aims to leverage current developments in the CPS field to analyze and control the interdependencies between computational systems, such as wireless sensor and actuator networks, and physical civil infrastructure.

1.4 DISTRIBUTED CONTROL ALGORITHMS FOR NONLINEAR CONTROL

Civil infrastructure systems are often spatially large and controlled in multiple ways and from multiple points of actuation. They sometimes exhibit nonlinear behavior and need to satisfy multiple and often contradictory objectives. Surprisingly, civil systems from multiple application areas can share very similar mathematical models. This thesis advances the control theory for a wireless CPS that can be applied to large-scale civil infrastructure systems exhibiting bilinear dynamical behavior. A model-predictive control (MPC) solution for bilinear systems is proposed and implemented in a wireless CPS framework applied to a hydronic system. Wireless control systems, such as the one proposed in this thesis, necessitate new control architectures since algorithms are executed via embedded computing and not in the traditional centralized controller. To this end, agent-based computing is naturally adopted in this work as an architecture for distributed computing problems and information sharing in wireless CPS.

1.4.1 Bilinear systems

Bilinear dynamical systems (BLSs) are described by differential equations in which the vector of dependent variables being differentiated is multiplied by another vector of known dependent variables. Equation (1.1) shows a generic BLS in a form similar to linear state-space systems, with an $N \times 1$ state vector $x(t)$, an $M \times 1$ control vector $u(t)$, and matrices A and B and vector b of appropriate size. Research into BLS has matured leading to multiple methods of analysis and control (Elliott 2007; Mohler 1970).

$$\dot{x}(t) = Ax(t) + \sum_{m=1}^{M} u_m(t)(Bx(t) + b) \tag{1.1}$$

BLS can be used to model common processes such as heat transfer, frictional retardation, and controllable viscous damping. In civil engineering these processes arise in the control of HVAC systems (Naidu and Rieger 2011; Ogonowski 2011; Oldewurtel et al. 2010; Piñón et al. 2005), structural vibration control (Elbeheiry 2001; Scruggs et al. 2007; Susumpow and Fujino 1995), wastewater treatment (Ekman 2005), and hydraulic networks (Zandvliet et al. 2007). This thesis investigates the model-predictive control approach applied to bilinear systems modeled using the bilinear model of (1.1) and a linear model with state dependent constraints. Both of these models are commonly found in the literature.

1.4.2 Model-predictive control

Feedback control of civil infrastructure requires a control law to maintain the desired system performance regardless of system nonlinearities (e.g. BLS characteristics), constraints on control authority, varying disturbances, and changes in system properties. Classical control methods such as proportional-integral-derivative gain control (PID) perform poorly for nonlinear systems and may not be robust to disturbances or model uncertainties. Adaptive controllers exist to adjust the controller gains as system properties or disturbances vary, and gain scheduling can account for control limits; however, both of these methods are difficult to realize for systems with fast nonlinearities. Model-predictive control (MPC) can tackle all of these challenges (Morari and Lee 1999). The MPC method updates the actuator output at a fixed period by solving for an optimal open-loop (OL) control trajectory predicted over a finite-time horizon. It does so by using a predictive model of the system embedded into the controller. This OL control problem is solved online at each actuator update period, which is possible because the OL trajectory optimization is often significantly less computationally complex than the closed-loop (CL) optimization required for classical control. The MPC can elegantly handle constraints, variances in future predicted disturbances, and system nonlinearities, including bilinear and hybrid dynamics as seen in (Del Re et al. 1993) and (Lunze and Lamnabhi-Lagarrigue 2009) respectively. This is achieved simply by utilizing the appropriate system model and numerical optimization parameters.

MPC was originally developed in the process controls industry, but is seeing more widespread usage as the theory and capabilities of microcontrollers improve (Camacho and Bordons 1999). MPC's ability to handle constraints and time varying future disturbances has been utilized to control HVAC systems subject to occupancy and weather changes (Aswani et al. 2012). Similarly, MPC has been proposed to control open water irrigation systems with stochastic models of future rain events (Maestre et al. 2012). Civil infrastructure control systems often need to be distributed, as will be expounded in more detail in later chapters, and distribution is an intrinsic capability of MPC (Camponogara et al. 2002). This thesis describes this and other uses of MPC for the control of civil infrastructure.

1.4.3 Hybrid dynamical systems

Hybrid dynamical systems (HDS) are modeled with continuous states described by differential equations and discrete states described by switching rules. Two comprehensive treatises that may be reference for modeling and control of HDS are (Lunze and Lamnabhi-Lagarrigue 2009) and (Goebel et al. 2009). Recent interest in HDS can likely be attributed to the accumulation of researchers from many applications with HDS behaviors under the umbrella of CPS. Systems that can be modeled using the HDS framework are found in many areas of civil engineering including semi-active structural control (Elhaddad and Johnson 2013), mining ventilation (Benedetto 2008), transportation systems (Cortés et al. 2010; Qin et al. 2009; Zhong and Sumalee 2008), air-traffic control systems (Livadas et al. 2000), and structural vibration control (Elhaddad and Johnson 2013). This thesis proposes that model-predictive control implemented on wireless controllers can successfully be applied to HDS with continuous states modeled by BLS.

1.4.4 Distributed control

Large-scale infrastructure systems present a challenging control problem. Traditionally the two options for control architectures were centralized or decentralized. Centralized architectures could achieve optimal performance, but were costly due to the required computational resources needed at the central controller and lengths of wiring to sensors and actuators. Decentralized architectures could be applied when the physical plant consisted of lightly interconnected subsystems with collocated sensing and actuation through autonomous controllers; however, these controllers were unable to communicate. Decentralization brought decreased installation costs, but also performance loss due to the limited local information each sub-system controller had about the global system status. In implementing the control system on a wireless CPS, a distributed control architecture can be employed by extending the decentralized architecture to include modest information sharing between 'neighboring' sub-system controllers.

This thesis will elaborate upon the use of this wirelessly enabled distributed model-predictive control architecture for controlling HDS with interconnected physical sub-systems, some of which include continuous BLS dynamics. The distributed control architecture utilized consists of an ‘agent’: an autonomous program that gathers information, senses, and actuates its surroundings to optimize some personal objective (Maturana et al. 2005). This agent is embedded within the wireless node controlling each subsystem. The wireless agent-based control (ABC) architecture proposed is applied and experimentally tested on a hydronics network test bed.

1.5 RESEARCH OBJECTIVES AND DISSERTATION OUTLINE

In this dissertation, a wireless controller framework is developed for a class of civil infrastructure characterized as a HDS with an interconnected physical plant characterized by bilinear continuous dynamics. A schematic of this architecture is shown in Figure 1.2. This framework fits within the cyber-physical systems paradigm that aspires to improve infrastructure performance, by accounting for the interactions of design constraints and objectives of the computational and physical subcomponents. The contributions of this thesis include a wireless hardware and software platform for research in control of cyber-physical infrastructure systems; new knowledge on model selection for MPC of BLS; and a method for wireless ABC of interconnected BLS with hybrid actuation. The resulting wireless CPS is experimentally tested and compared against benchmarks. The outline of the thesis is now delineated.

In *Chapter 2: The Martlet Wireless Platform: Enabling Cyber Subsystems* background information is provided on the portions of wireless networking theory that are applicable to wireless control of cyber-physical infrastructure. Deficiencies in the control capabilities of wireless nodes in the current market are highlighted and a new wireless node is developed, named the *Martlet*, and shown to meet the needs of the wireless CPS research community. The *Martlet*

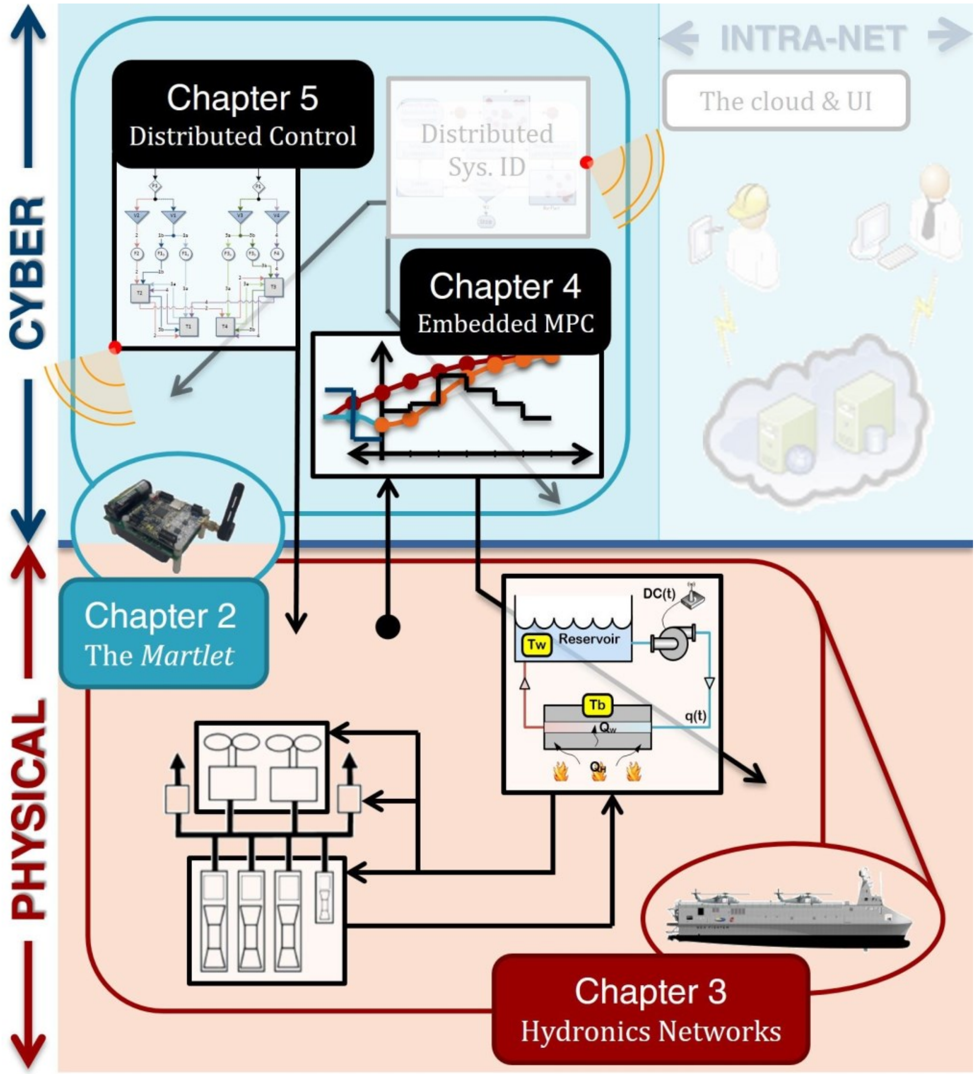


Figure 1.2 Thesis overview

is tested and validated as a viable tool for infrastructure monitoring through an extended deployment measuring vibrations on a wind turbine tower in Los Alamos, New Mexico.

In *Chapter 3: CPS Paradigm Applied to Hydronic Cooling Infrastructure* interconnected hydronic systems are described in a variety of infrastructure engineering applications, including new, highly-integrated naval vessels. An experimental test bed is developed as a representation of shipboard cooling networks for the purpose of experimental verification of wireless CPS controllers, including hardware and software, for hydronic cooling applications.

In *Chapter 4: Bilinear Systems (BLS): Continuous Dynamics* and Control*,* additional information is provided on bilinear systems detailing their ability to model a variety of phenomena in infrastructure engineering applications, including heat transfer in hydronic systems. Two common approaches for modeling BLS control systems – linear model with state-dependent control constraints and bilinear model with time-invariant rectangular control constraints – are studied to determine their applicability for controlling wirelessly controlled hydronics networks. Multiple objective functions for hydronic cooling are analyzed along with their influence on the cyber and physical subcomponents of the proposed wireless CPS. An algorithm that fits the CPS constraints is presented which solves the OL control problem posed for hydronic systems. The BLS model is selected, along with one of the objective functions, for incorporation into an MPC that is embedded into a *Martlet*. The control law is autonomously executed, and promising experimental data on a simple hydronic system is described.

In *Chapter 5: Hybrid Dynamics and Distributed Control of* CPS, background and analysis is provided on the cyber-physical interactions of bilinear systems, model-predictive control, hybrid dynamics, wireless control, and agent-based distributed computation. An ABC is proposed for the control of networked hydronic systems with hybrid dynamics when subject to the constraints and interdependencies of wireless CPS. The agent-based control (ABC), implemented on a network of *Martlets*, is shown to outperform benchmark controllers when controlling the hydronic test bed during a comprehensive test scenario.

Lastly, in *Chapter 6: Conclusions and Future Directions*, highlights and key contributions of the thesis are summarized. Additionally a framework is introduced for future work that characterizes the challenges and opportunities ahead in the area of wireless CPS.

Chapter 2.

THE *MARTLET* WIRELESS PLATFORM: ENABLING CYBER SUBSYSTEMS

To create a cyber-physical system (CPS), sensors and actuators are installed in the physical system being monitored or controlled. Wired versions of sensors and actuators have possibly slowed the emergence of CPS due to the cost and complexity of installing the wired infrastructure. On the other hand, wireless telemetry is opening new opportunities for the implementation of CPS. This chapter introduces a wireless platform to enable wireless sensing and actuation. The node developed embraces the need for computing in CPS frameworks by supporting a dual-core computing element for local data processing and control law execution. The new wireless platform has been named the *Martlet*[1]; see Appendix A for a quick-reference datasheet.

The precursor to the *Martlet*, the *Narada* wireless sensing unit (Swartz et al. 2005) shown in Figure 2.1, has been a very successful platform for developing monitoring systems to track the structural performance of civil engineering (Kim and Nadukuru 2012; Kurata et al. 2012), and non-civil engineering structures (R. Swartz, Zimmerman, et al. 2010). The *Narada* platform has also been shown to be capable of distributed data processing for system identification (Kim and Lynch 2012; Zimmerman and Lynch 2009), controlling the response of buildings subject to earthquakes (Kane, Lynch, and Law 2011; Swartz and Lynch 2007, 2009; R. A. Swartz et al. 2010; Wang et al.

[1] The wireless device was named after the *Martlet* heraldic charge, depicted as a small fast bird which symbolized the fourth son, virtue, and adventure (Coats 1747). The wireless device contains a fast 80 MHz processor, is capable of learning and actuation, and sits as the 4th prototype in line starting with Stasser and Kiremidjians WiMMS device in 1998 (Straser et al. 1998), the 2nd generation WiMMs devices in 2001 (Lynch et al. 2001), and the *Narada* in 2005 (Swartz et al. 2005).

2009), and monitoring and control of industrial processes (Kane and Lynch 2012). Key amongst the *Narada*'s strengths were a relatively simple set of source-code, ad-hoc communication capabilities, reliable data acquisition, and simple interfaces to sensors and actuators.

However, its design had a number of limitation including its non-trivial power management sub-system, connection issues with sensors and actuators in high vibration environments, and a slow processor without native floating point calculation capability. With these strengths and weaknesses in mind, the development of the *Martlet* began as a collaboration between the Laboratory for Intelligent Systems and Technologies (LIST) at the University of Michigan (Ann Arbor, MI), the Laboratory for Smart Structural Systems at Georgia Institute of Technology (Atlanta, GA), and the Department of Civil and Environmental Engineering at Michigan Technological University (Houghton, MI). The primary goal of the *Martlet* development effort was to create a new wireless device, the likes of which did not currently exist in academia or the commercial market, which featured:

- wireless communication, preferably backwards compatible with the *Narada;*
- a faster processor capable of hardware floating point calculations;
- extensibility through the addition of sensors, actuators, and signal conditioning peripherals with strong mechanical connections and many options for electrical interface;
- JTAG debugging capability that would reduce development time for new applications;
- and suitable memory for data processing and complex algorithms.

The *Martlet* design process is documented in this chapter. This begins with an overview of wireless sensor and actuator networks and their application to CPS. This discussion is followed by a description of the

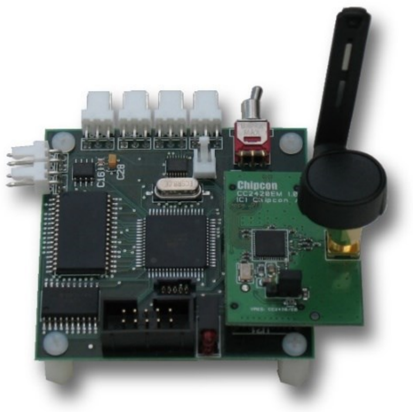


Figure 2.1 The *Narada* wireless node

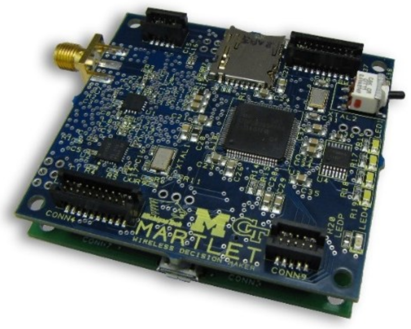


Figure 2.2 The *Martlet* wireless node

embedded software in the *Martlet*. Next, the successful design of the *Martlet* was field tested in a structural monitoring role which is detailed. The chapter concludes with a summary and overview of the *Martlet*'s value in enabling wireless CPS. This summary also serves as an introduction into the subsequent chapters in which the *Martlet* was used for executing nonlinear control algorithms to control a chilled water hydraulic network cooling thermal loads.

2.1 WIRELESS NETWORKING PRINCIPLES FOR INFRASTRUCTURE SYSTEMS

Methods of data transfer between sensors, actuators, data-processors, and data-storage, (i.e. the ‘cyber’ components in a CPS) can be classified into one of two categories: wired, or wireless. Regardless of the method, network management practices must be established in order to ensure reliable and efficient communication. These practices are especially important in a radio frequency (RF) wireless digital network since nodes share a limited frequency spectrum to transmit and receive data. This is in stark contrast to wired networks where each node can have a direct link and often dedicated link to the data aggregator. Before considering the network practices as a whole, it will be helpful to first consider the communication link between two nodes since peer-to-peer connectivity is the fundamental building block upon which wireless mesh networks are built.

Table 2.1
Comparison of Wired and Wireless Communications

	Source	Transmitter	Channel	Receiver	Destination
Wired monitoring system	Transducer		Co-axial cable (1 ch / wire)	Data acquisition (ADC)	Server
Wireless monitoring system	Transducer	ADC + radio	RF medium *(multiple ch in spectrum)*	Radio	Server

The mathematical theory of communications for designing network management practices was arguably founded by Shannon in (Shannon 1928) in which he defines the five fundamental components of a communication link. Table 2.1 provides an example of the five components for a wired and wireless monitoring system.

1) The *information source* generates a message and desires to communicate it to the receiver. Commonly, the information to be transmitted in wireless CPS is the processed sensor and actuator data.

2) The *transmitter* transforms the message into a signal to be sent over the *channel*. In a wireless sensor network (WSN) the *transmitter* converts the transducer's electrical signal into a digital value using an analog-to-digital converter (ADC), buffers and packetizes the digital readings, and modulates the digital data onto part of the RF spectrum

3) The *channel* is the medium over which the message is transmitted. Wireless systems transmit data on a RF over whatever medium is between the *transmitter* and *receiver,* whether that be air, soil, water, or building walls. The signal is degraded along the *channel* by attenuation and electromagnetic (EM) interference such as other wireless networks, microwave ovens, and fluorescent lights (Cisco Systems Inc. 2007).

4) The *receiver* acts in the opposite manner as the *transmitter*. On a wireless node, the receiver decodes the noisy and attenuated signal obtained from

the *channel* and recovers, as close as possible, the digital message that the *source* desired to send.

5) The *destination* is the device for which the message was intended. The ultimate destination for most messages in a monitoring system (wired and wireless) is the site's data aggregator. However, in a wireless network a message may have to 'hop' from one node to another if the wireless signal is not strong enough to reach from the *source* to the final *destination*.

Communication networks consist of these source-transmitter-channel-receiver-destination links that interconnect the nodes in the network. Network nodes produce, process, route, and act on the communications. The Open Systems Interconnection (OSI) reference model provides a means for abstracting a network design into seven hierarchical levels (Zimmermann 1980). The OSI reference model was created by the *Organisation Internationale de Normalisation* (ISO) subcommittee on OSI in 1977 as a general framework to build network hardware and *stacks*, the network protocol software. The OSI reference model layers start at the most fundamental *physical layer* and grow to become more abstract, terminating at the *application layer*. Below is a brief description of each layer, and its key relations to wireless networking for infrastructure systems.

1) The *physical (PHY) layer* defines the method by which data is transferred over the physical media. In RF wireless digital communications one of the simplest methods of physical data transfer is on-off key (OOK) modulation in which a carrier frequency amplitude is switched to transfer 0- and 1-bits. More robust modulation techniques, such as direct sequence spread spectrum (DSSS), are used in networking standards like IEEE 802.15.4 (IEEE Computer Society 2006). The selection of a carrier frequency, such as 900 MHz versus 2.4GHz, will affect data rate (↑ frequency ⇒ ↑ bandwidth), range (↑ frequency ⇒ ↓ range, especially through solid obstacles), and amount of interference. If multiple networks are in the

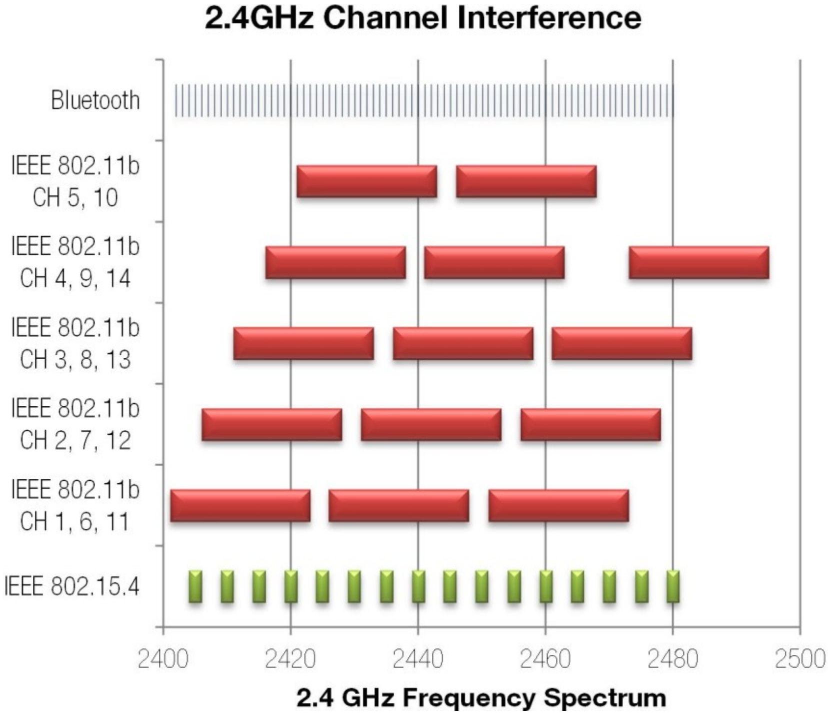


Figure 2.3 2.4 GHz channel-frequency interference

same vicinity, carrier frequencies should be selected or changed to prevent overlap of the finite spectrum used by each carrier frequency. Figure 2.3 shows such overlap for collocated Bluetooth, IEEE 802.11b, and IEEE 802.15.4 network channels.

2) The *data link layer* establishes the link between nodes in the network using an addressing system and defines how addresses are attached to each message's frame. In the *Martlet*, and other networks using the IEEE 802.15.4 standard, the data link layer is comprised of two sub-layers. The upper logical link control (LLC) sub-layer shields upper layers of the stack from the specifics of the underlying physical layer. The lower media access control (MAC) sub-layer establishes when nodes are allowed to transmit on the channel. This could be defined by a schedule-based protocol such as time-division multiple-access (TDMA) or a contention-based protocol such as clear-channel assessment (CCA). The *Martlet* is capable of TDMA and/or CCA communication depending on the application, and sends all of the MAC information in the packet structure shown in Figure 2.4. The packet

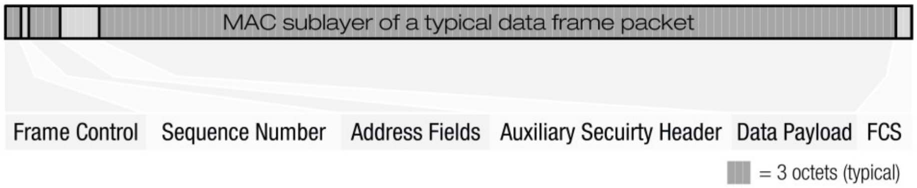


Figure 2.4 Schematic view of the IEEE 802.15.4 Data Packet MAC sub-layer (IEEE Computer Society 2006)

contains the source and destination address, packet sequence information, security details, a payload of up to 102 bytes, and a check sum to ensure data integrity when received (IEEE Computer Society 2006).

3) The *network layer* provides routing and switching functionality. For wireless networks, this layer is built around one of the three main topologies shown in Figure 2.5: star, mesh, and hierarchical tree topologies. Star networks offer high data rates to a central coordinator; however all nodes must be within range, and robustness is decreased due to the centralized point of failure. Mesh networks replace the coordinator in favor of decentralized peer-to-peer links and enable mobile nodes and redundant communication links, via multi-hop paths. Hierarchical topologies permit efficient use of heterogeneous networks with low-power leaf nodes and higher-power, more-capable trunk nodes relaying messages at higher data-rates, often on another frequency (Kottapalli et al. 2003; Kurata et al. 2012). All topologies can be implemented on the *Martlet* by the user but the base configuration relies on a single-hop peer-to-peer network layer.

4) The *transport layer* provides transparent and reliable data transfer to the upper protocol layers. The important wireless communication tasks of error control and failed message retransmission are handled by this layer. The transport layer in *Martlet* networks was designed as a simple byte-stream that the application must know how to compose and parse.

5) The *session layer* controls the binding and unbinding of sessions, i.e., brief amounts of time in which the physical layer is dedicated to data-transfer

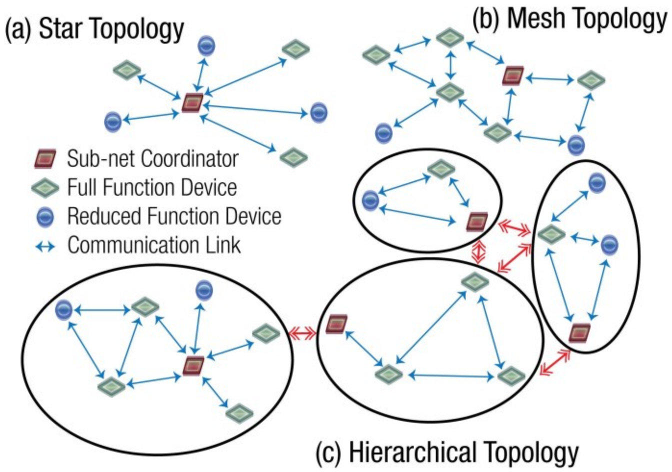


Figure 2.5 Typical WSN topologies
(Adapted from (IEEE Computer Society 2006))

between two specific nodes. Since the *Martlet* network stack was designed to be as small as possible in order to fit on a wireless sensor node's limited memory, the data transfer session is comprised of only a single packet. Larger amounts of data must be sent in multiple packets handled by the application or by network middleware (see section 2.2.3).

6) The *presentation layer* creates an abstracted interface to the layers below, so higher-level applications may be written regardless of the type of underlying network used. The *Martlet*'s presentation layer, which was modeled after and made backwards compatible with *Narada*'s, consists of a data structure containing the packets' source, the packet length, a byte-stream of data, and a command byte used by the application to choose the appropriate parser for the data byte-stream.

7) The *application layer* should be the only layer of the OSI reference model that the user's applications interacts with. This is implemented on the *Martlet* in a series of functions for requesting data, sending packets, handling acknowledgements, and receiving packets.

The specification of these seven layers is important for wireless sensor and actuator networks. These decisions should include consideration of the desired data rate, communication range, and power consumption. The

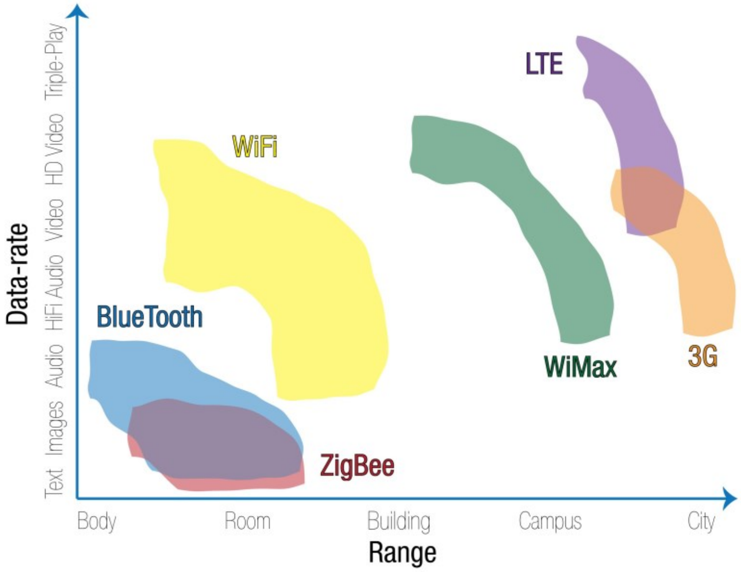


Figure 2.6 Range and data rate of wireless standards
(Adapted from (Sohraby et al. 2007))

qualitative comparison of wireless networking standards in Figure 2.6 and Table 2.3, indicate IEEE 802.15.4 based networks are a common architecture among deployments of wireless sensor networks (Lynch and Loh 2006). Designers can either select a fully defined standard architecture such as ZigBee (ZigBee Alliance 2014) or WirelessHART (Chen et al. 2010), select common components that fit together such as WiFi based TCP/IP secure HTTP, or fashion their own architecture based off of a partial standard like IEEE 802.15.4 to meet system constraint as was done with the *Narada* and *Martlet* in order reduce latencies. Such a breakdown of design responsibilities is shown in Table 2.2. With IEEE 802.15.4 physical and data link layers, the *Martlet* stack adds on a mesh networking layer, a transport layer with cyclic redundancy checks (CRCs) and acknowledgements (ACKs) to mitigate data corruption and loss, and a single-session byte-stream that is presented to the application layer through function calls for requesting data, sending packets, and receiving packets.

The *Martlet's* network stack, described in Table 2.2, was chosen to meet the design requirements set out at the beginning of this chapter. The key constraint was backwards compatibility with the *Narada* and other popular

Table 2.2
Common Network Stacks and Development Responsibility

OSI Stack Layer	ZigBee®	*Martlet*	WiFi™
User Application	User defined		
(7) Application Layer	ZigBee Alliance	User defined	FTP or HTTP
(6) Presentation Layer		Byte-stream*	MIME
(5) Session Layer		Single-session*	SSL
(4) Transport Layer		CRC with ACK*	TCP or UDP
(3) Network Layer		Mesh*	IPv4 or IPv6
(2) Data Link Layer	IEEE 802.15.4		IEEE 802.11
(1) Physical Layer			

*Custom designed by *Martlet* designers, modeled after *Narada*

wireless nodes such as the iMote by Memsic. However, the *Martlet* team would have considered another stack if the benefits of switching out-weighted the loss of backwards compatibility. A ZigBee stack was not pursued because of its significant latency and packet timing jitter due to overhead in upper layers of the stack. WirelessHART, an industry standard for wireless control, was strongly considered, but ultimately not selected because the high precision crystals oscillators consume significant power. Also restricting the use of WirelessHART was the TDMA data link layer's relatively long period which would limit feedback control rates. Additionally, none of the wirlessHART development kits on the market in 2010 enabled ad-hoc communication. These drawbacks were unable to overcome the convenience of utilizing an industry standard (i.e. ISA100) network stack, leading the team to settle on the continued use of a stack similar to that used by the *Narada*.

Table 2.3
Qualitative Comparison of Wireless Standards

OSI Layer	IEEE 802.15.4	BlueTooth	IEEE 802.11
Power consumption	Ultralow	Low	Medium
Battery life	Days to years	Hours to days	Minutes to hours
Cost & complexity	Low	Medium	High

Adapted from (Peters 2005)

2.2 HARDWARE DESIGN

Data systems for cyber physical infrastructure systems can be decomposed into four main parts: the data acquisition system (Figure 2.7d through g), the actuation system (not pictured), the backend data management and analytics system (Figure 2.7b), and the user frontend (Figure 2.7a). The developments made by this dissertation contribute to driving the computational load from the computers in the backend down into the sensing and actuating wireless nodes.

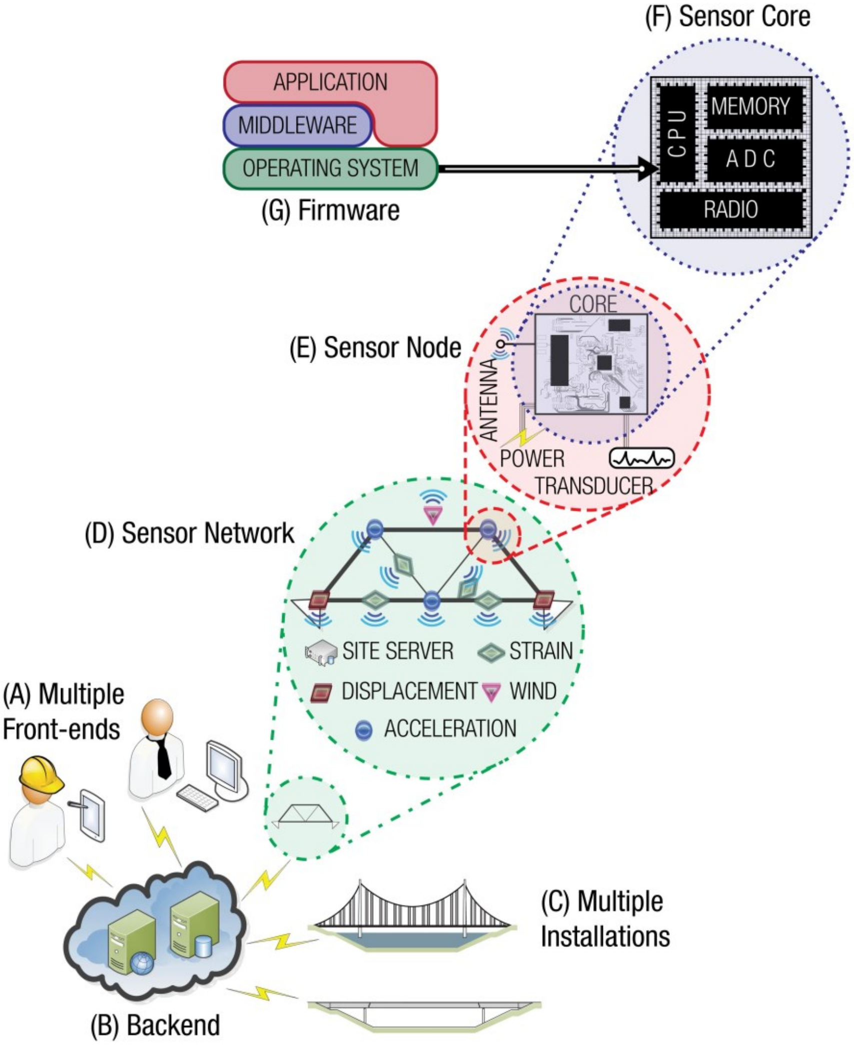


Figure 2.7 Overview of the wireless structural monitoring system architecture

2.2.1 Anatomy of a wireless node

Fundamentally, wireless nodes require a processing unit, typically a micro-controller (MCU), interfaced with a digital radio. The MCU is essentially an extremely small computer that processes measurements, generates control signals, buffers data, and handles packetized data to and from the wireless transceiver. In order to execute these tasks, the computational core typically has the following features, which can be packaged into a single system-on-a-chip, or implemented in separate integrated circuits (ICs) interfaced to the MCU:

- Volatile memory (RAM) for temporary data storage
- Non-volatile memory, namely EEPROM or FLASH, for storage of programs, calibration values, and unit information
- Analog-to-digital converters (ADC) for sensing
- Digital-to-analog converts (DAC) for control
- Serial interfaces, namely SPI, UART, and I2C, to talk to external chips such as external memory, radios, ADCs, and DACs
- Coprocessors for efficient floating point calculations, signal processing, encryption, or control laws
- Timers and counters for system timing, event counting, and digital signal generation
- Watchdog timer to prevent system hang-ups
- Programming and debugging support

The key architecture choices for embedded processors are operating voltage, bus width, clock speed, memory type and size, and integrated peripherals. Since wireless devices often run off of batteries and/or limited harvested energy, thus removing the ‘last wire’, power consumption must be minimized. The most common supply voltages are 5V and 3.3V but more 1.8V devices are coming to market since lower operating voltages generally correlate to lower power consumption. The lowest power MCUs, such as the

ATMega128 used in the *Narada*, have 8-bit wide data buses. This means that only 8-bit integer math can be performed in a single clock cycle. Multiple clock cycles are required in order to execute 16-, 32-, or even 64-bit calculations. Increasing the bus width plays the tradeoff between power-per-clock-cycle with power-per-computation. Additionally, 16-bit and higher MCUs may have the ability to complete floating point calculations in a single clock cycle. This is important for digital signal processing. Processors are available with clock rates from less than 1MHz to greater to 1GHz. Increasing the rate increases instantaneous power consumption. On the other hand, increasing the clock-rate may enable either more complex real-time calculations or, through the use of low-power sleep modes, a more efficient design in terms of power-per-calculation as the processor is only 'awake' during shorter periods of time.

More and more MCU manufacturers are integrating additional peripherals such as ADCs, larger amounts of memory, and even wireless radio circuitry into their MCU product lines. Each of these can drive down circuit component counts, thereby reduce node costs and simplifying design. However, off-chip peripherals may still be desired if higher performance is required, for example a higher resolution ADCs.

Wireless transceiver modules, as opposed to customized radio circuits, simply node design by reducing the need for RF engineering through firmway that implement lower layers of the RF stack. Modules are available that operate on the 900 MHz or 2.4GHz industrial, scientific, medical (ISM) license-free spectrum implementing IEEE 802.15.4 (ZigBee, *Martlet*, *Narada*, iMote, etc.) IEEE 802.10 (Bluetooth), and IEEE 802.11 (WiFi) standards. The radio module often consumes more power than the rest of the components on the node, therefore receiving and transmission power consumption are key selection criterion, balanced with transmission range and receiver sensitivity. Proper antenna selection can increase communication reliability and range without increased power consumption. 'Smart antenna' designs which can actively control output power and signal direction are an active area of research (Özdemir et al. 2010). Radio power consumption can be further reduced by sleeping the radio when a precisely timed TDMA protocol is used in the data link layer.

2.2.2 Peripherals

Besides the peripherals embedded into the node core (Figure 2.7f) additional peripherals such as transducers, signal conditioners, energy harvesters, and user interfaces must be supplied to interface the computational core with the physical world.

Wireless sensors can measure most of the same phenomena as their wired counterparts, albeit sometimes in a different way as to minimize power consumption. For example, when measuring acceleration, inexpensive small low-power MEMs accelerometers are most often used, as opposed to more accurate force-balance accelerometers. This aligns with the trend seen wireless CPS, that many inexpensive sensors can outperform a few sensors providing high-grade signals.

Regardless of the type of sensor used, except for integrated digital sensors, signal conditioning circuitry is required to interface the signal with a node's core. Simple sensors such a metal foil strain gauges, need to be interfaced with circuits like a Wheatstone Bridge which convert the sensors change in electoral properties into a voltage-proportional signal within the range of the ADC. Many sensors are now integrating this analog signal conditioning, in which case only an amplifier and anti-aliasing filter is required between the sensor and ADC. New digital signal processing techniques enable large analog circuits to be replaced with mixed signal ICs such as the QuickFilter QF4A512 (Quickfilter Technologies 2009). This IC has seen successful usage in the iMote2 'SHM-H sensor board' (Jo et al. 2012) used in civil structural monitoring applications.

Actuators that close the sensing-computing-control feedback loop inherent in CPS are the often the largest power consumer. Therefore if a low-power wireless system is desired efficient actuators must be selected. For example, force can be applied to a vibrating structure with just a few Watts with a magnetorheological damper instead of a hydraulic actuator requiring hundreds of Watts (Housner et al. 1997). If energy consumption is not so much of an issue, a wireless controller can be interfaced with power amplifiers in the same way as a traditional programmable logic controller (PLC) would be.

From this observation of energy consumption comes the general rule of design for wireless CPS: **Computation is cheap, actuation is expensive, and sensing lies in between;** therefore utilize complex calculations to reduce the amount of sensing and actuation required, and similarly increase the sensor density if doing so reduces actuation requirements.

Removing the ‘last wire’, that is the power wire, from a wireless CPS requires mindful selection of components as discussed above, but also an ability to harvest energy from the environment and store it in an efficient way. Table 2.4 lists the energy density of select harvesting and storage technologies for civil infrastructure applications. It should be noted that these values are typical, and load and environmental conditions can decrease performance. For example large loads and winter temperature can significantly decrease battery energy capacity. If the WSN designer must design a custom energy harvesting solution, reference texts such as Beeby and White's *Energy Harvest for Autonomous Systems* (Beeby and White 2010) provide details on each mode of energy harvesting and design guidelines. By using energy efficient wireless sensors and an appropriately designed energy harvesting and storage system, WSN can be deployed maintenance free for years.

Table 2.4
Energy Densities of Common WSN Power Sources

Energy Harvesters	Power Density	Battery Chemistry	Volumetric Density ($^{WHr}/_{L}$)
Solar (outdoors)	15 mW/cm2 (sunny) (Kurata et al. 2012)	Alkaline-Mn02*	347
	0.15 mW/cm2 (cloudy)	Sealed lead acid	90
Solar (indoors)	0.006 mW/cm2 (ambient light)	NiCd	80-105
	0.57 mW/cm2 (task light)	NiMH	175
Vibrations	0.01-0.1 mW/cm3 (Mccullagh et al. 2012)	Li-ion	200
Acoustic noise	3E-6 mW/cm2 @ 75 dB 9.6 mW/cm2 @ 100 dB	Li-Polymer	300-415
Applied RF	2-8 mW (Mascarenas et al. 2010)		* Non-rechargable Adapted from (Vieira et al. 2003)

2.2.3 The *Martlet* hardware design

For the *Martlet* to excel in monitoring and controlling infrastructure systems, it must have low power consumption, low latency, and the ability to quickly process data and execute control algorithms. Striking a balance between the low-power, yet insufficient speed, of 8-bit MCUs and the unacceptable power consumption of 32-bit MCU, the *Martlet* was designed around a 16-bit Texas Instruments (TI) TMS320F28069 modified Harvard Bus Architecture MCU with on-the-fly programmable clock rates up to 80 MHz. Real-time digital signal processing is possible with native single-precision floating-point calculations. The Viterbi, Complex Math, CRC Unit (VCU) extends the instruction set to support complex multiplication, Viterbi operations, and Cyclic Redundancy Checks (CRCs). The computationally intensive tasks of control processing are offloaded from the main CPU and onto a programmable Control Law Accelerator (CLA) 32-bit floating-point math accelerator where precise timing can be more easily achieved. A 9-channel dual sample-and-hold 12-bit ADC is capable of collecting signals from transducers at up to 3 MSPS. Programs residing in the 256 kB x 16-bit flash memory can process this data, or the 6-channel Direct Memory Access (DMA) controller can directly feed the ADC data into the 100 kB x 16-bit random access memory (RAM) for further processing by the CLA. If additional data storage capacity is required, a microSDHC card can be inserted into the on-board card reader giving the *Martlet* up to 32 GB of additional flash memory. A variety of General Purpose Input Output (GPIO) pins are available to produce control signals through communication protocols such as UART, SPI, I2C, and/or pulse width modulation (PWM). The fast interrupt response inherent in the architecture of the MCU together with three 32-bit timers makes this MCU an excellent choice for control applications with strict timing requirements.

A radio transceiver working on a free ISM frequency band extends the decision making capabilities of the *Martlet* by enabling a flock of *Martlets* to communicate amongst themselves and generate decisions which are best for the control system as a whole. The *Martlet's* transceiver is formed by pairing a TI CC2520 2.4 GHz IEEE 802.15.4 transceiver with a TI CC2591 RF Front End which adds a programmable low noise amplifier (LNA) for improved receiver sensitivity and power amplifier (PA) for improved output power. The MCU can control transceiver power consumption by varying output power from 55mA at -8 dBm to 55mA at +17 dBm, varying receiver sensitivity from 23 mA at -50 dBm to 26mA at -90 dBm, and through the use of sleep modes drawing as little as 1µA. With appropriate power management firmware in the MCU, *Martlet* power requirements will be significantly less than competing devices on the market, yet with equal or improved communication range.

Inspiration for the new capabilities featured on the *Martlet* come from years of experience with the design and use of wireless devices for civil engineering applications. The *Martlet* printed circuit board (PCB) features 16 different test points giving developers access to all intra-board signals, and eavesdropping devices can be installed on inter-board signals to aid in the development of peripheral layers. Even though the PCB will be installed in protective enclosures while deployed in the field they still experience significant mechanical stresses and vibration. Peripheral layers are securely joined with threaded plastic standoffs preventing layer separation. The use of edge mounted SMA antenna connectors soldered to both sides of the PCB and the use of through-hole posts on inter-board connectors will mitigate the possibility of premature connector failure. Sleep or power enable pins have become a standard addition on many integrated circuits (ICs) produced today. In order to take advantage of this new industry standard feature the *Martlet* provides GPIO pins to the developer to tie to these power management pins giving software power to be as frugal as possible.

The key to the *Martlet's* success in enabling CPS research will be its ability to sense and control a multitude of different systems and physical properties. Initial applications for the *Martlet* include vibration analysis of wind turbine structures, semi-active structural control of bridge-structures, hydronic system monitoring and control, and HVAC monitoring and control. The *Martlet* is able to sense and control these systems because of its extensible design. *Martlet wings*, the name given to the boards mounted above and below the *Martlet*, have already be created for a wide variety of purposes: interfacing with transducers measuring strain, acceleration (Figure 2.8), ultrasonic transducers, fluid flow (Figure 2.9), CO_2, and temperature (Figure 2.10); accurate time keeping with a real-time clock (RTC) (Figure 2.15); band-pass filtering signals with high dynamic range by means of programmable filter gains and cut-off frequencies (Figure 2.14); actuating MR dampers (Figure 2.14), motors (Figure 2.11), pumps, and valves (Figure 2.9); programming and debugging (Figure 2.12); managing and supplying power; and prototyping (Figure 2.13). These *wings* have been designed relatively quickly and easily using standard signal conditioning and interface circuitry found in texts such as (Horowitz and Hill 1989), and laid out on standard *wing* templates. The potential to utilize the *Martlet* for differing CPS interfaces is nearly limitless due to flexibility of the *wings* to communicate with the *Martlet* core via analog signals into the *Martlet*'s ADC; or via digital protocols including SCI, I^{2}C, UART, or GPIO; and receive power from or supply power to the *Martlet*.

2.3 Software design

There are two main parts to wireless CPS software: the embedded firmware running on the wireless nodes, and software for data presentation, processing, and archiving. This section will focus on the embedded firmware which faces new constraints due to the wireless CPS paradigm. The firmware can be broken down into three layers: (Figure 2.7g): the operating system (OS), middleware, and application software. Ideally, a hardware abstraction layer (HAL) is provided to create a level of abstraction between the hardware and OS, and an application programming interface (API) is provided to abstract OS and middleware from the application. These layers and abstractions enable hardware flexibility, greater amounts of code reuse, reduced development time, and decreased code maintenance costs.

2.3.1 Operating systems

The software that makes up the operating system (OS) on a wireless node provides system management features and hardware interfaces to the upper middleware and application layers. It should be noted that the OS on a wireless node is significantly different from the OS on a consumer PC running Windows, Mac OS, or Linux, due to the limited memory and computational capability. In addition to the computational constraints, embedded OS must have methods to conserve power and quickly respond to external events. The OS can save power by putting the node into a sleep when all tasks in the queue have completed, and wake again when an interrupt occurs. Interrupts also enable preemptions of a currently running thread of low priority by a task of higher priority. Task preemption enables real-time response, but can lead to system failures like deadlock and priority inversion, which a well-designed OS and API can mitigate using strategies outlined in (Coffman et al. 1971). The

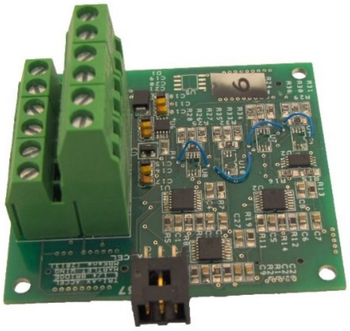


Figure 2.8 *Martlet* strain & acceleration *wing*

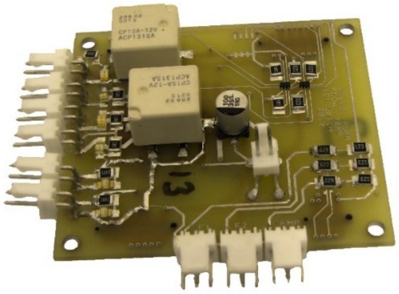


Figure 2.9 *Martlet* hydronics *wing*

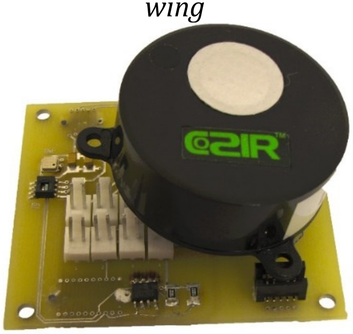


Figure 2.10 *Martlet* HVAC *wing* (designed by Mitch Hirose)

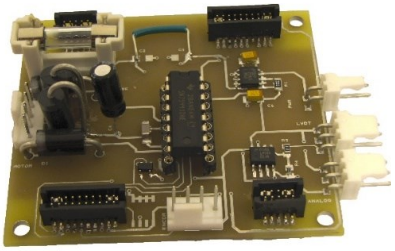


Figure 2.11 *Martlet* motor *wing* (designed by Courtney Peckens)

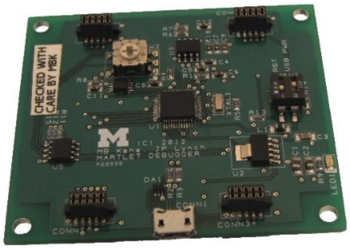


Figure 2.12 *Martlet* programming + debugging *wing*

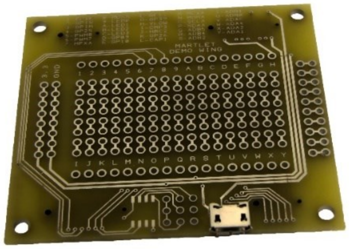


Figure 2.13 *Martlet* breadboard *wing*

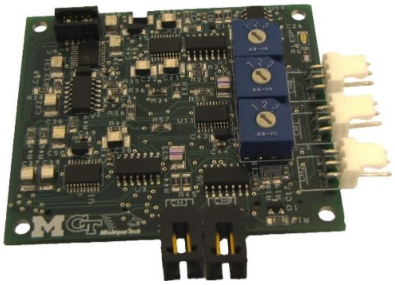


Figure 2.14 *Martlet* smart filter *wing* (designed by Dapeng Zhu)

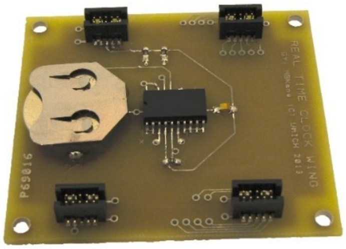


Figure 2.15 *Martlet* RTC *wing*

later fault famously led to a failure of the NASA Mars Pathfinder spacecraft; fortunately the error was discovered and fixed through a remote system upgrade (Reeves 1998).

The ubiquity of embedded systems has led to the development of a wide range of embedded OSs. Real-time operating systems (RTOSs) are multi-threaded OSs that thread preemption within specified time constraints and should autonomously handle common system failures. TinyOS (TinyOS Alliance 2012) is a single-threaded non-real-time operating system that was developed specifically for wireless sensor networks. Extra care must be taken when using non-RTOSs such as TinyOS to ensure time-critical tasks such as data collection and control are synchronized across the network. The complexity, expense, and memory requirements associated with an OS is not always required, in which case a simple interrupt based state machine along with a HAL can be custom developed. This strategy has been successfully employed on the *Martlet* and *Narada* wireless nodes where a state-machine creates a multi-threaded framework for real-time operation.

2.3.2 Middleware

Middleware exists to extend the capabilities of the OS and provide greater functionality to the applications. Examples of middleware include boot loaders, resource allocation, data compressors, synchronization tools, and network stack extensions.

Wireless boot loaders enable remote firmware upgrades of each node, thus reducing costs from manual upgrades or costs from buggy firmware. Boot loaders also enable software 'agents' to be dynamically distributed throughout the network (Fok et al. 2005; Spencer et al. 2004) as resources become available on each node. Novel techniques for resource allocation include: adaptive fidelity algorithms, in which nodes near an 'important event' sample with greater resolution than those far away (Estrin et al. 1999); and dynamic allocation of computation resources using a buyer/seller framework

(Zimmerman et al. 2009). Energy and bandwidth can be saved by reducing the number of bits wirelessly transmitted by compressing the data to be sent using a lossless compression algorithm (Lynch et al. 2003).

Synchronizing all the nodes in the network is an inherent challenge in wireless networks because each node has a separate clock. De-synchronization can occur on the network level due to network latency and on the hardware level due to gradual drifts of a node's clock. In star network configurations (see Figure 2.5), a coordinator can send out a single beacon to synchronize all the nodes in the network. This technique has been used successfully to maintain clocks with 30µs and total latencies of less than 10ms (R. Swartz, Zimmerman, et al. 2010). In multi-hop mesh and hierarchical network topologies the flooding time synchronization protocol (Maróti et al. 2004) provides a means of synchronization that quantifies the stochastic delay in each link of the network. Synchronization is required for precisely timed data acquisition and for energy efficient TDMA network protocols. A TDMA data-link layer can save energy by sleeping the units except for during the window when node is either set to send or receive data. Multi-hop message routing middleware, which extends the network layer of the network stack, can save energy through reduction in the required radio output power. The area under the power curves in Figure 2.16 represents the power required for each 'hop'. If n transmissions are used to cover a range R, then the reduction in total transmission power is proportional to R^2/n. However, in practice, choosing the correct transmission power, if variable at all, is a difficult task to accomplish reliably. Additionally, the power savings need to balance the increased latency associated with each 'hop' and the exponential decrease in reliability with respect to the number of 'hops', which would increase power consumption due to retransmissions.

The most extensive middle package for structural monitoring was developed by Nagayama, et al. (Nagayama et al. 2009) for the iMote2 running TinyOS and includes features such as reliable data transfer, network data

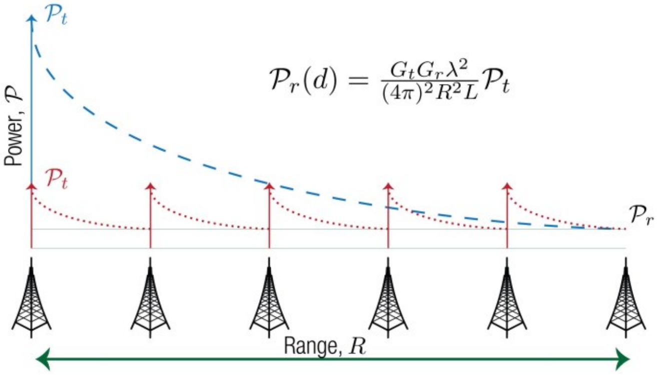

$\mathcal{P}_t$
$\mathcal{P}_r(d) = \frac{G_t G_r \lambda^2}{(4\pi)^2 R^2 L} \mathcal{P}_t$
Power, $\mathcal{P}$
$\mathcal{P}_t$
$\mathcal{P}_r$
Range, R

Figure 2.16Power usage in multi-hop networks

aggregation, and sensor synchronization. Outside the realm of monitoring civil structures, middleware has been developed for a wide variety of deployments, including body sensor networks (BSN), robot teams, and sensor networks to enable the 'smart' electric grid. A novel wireless BSN application reduced energy consumption partially through intelligent use of the RF spectrum and clock synchronization (the cyber part of CPS), and partially through adapting the requirements on information transmission depending on the nodes perceived condition of the body (the physical part of CPS) (Calhoun and Lach 2012). Middleware for mobile robot networks have been developed for multi-hop strategies when the network topology is dynamically changing (Fink et al. 2012). The smart electric grid needs middleware for securing communication between nodes, ensuring end-user privacy, and preventing compromise of the system through cyber- or physical-attack vectors.

2.3.3 Application software and user interface

The application software embedded into wireless nodes is significantly different than the applications that run on a typical PC. Even though the global systems computation and memory resources might be equivalent to a PC, they are distributed across all the network as schematically shown in Figure 2.17. The global design effectively is a multi-threaded system where single threads run on each node at any given time. The distributed embedded application must be efficient in terms of processor cycles, memory, network bandwidth,

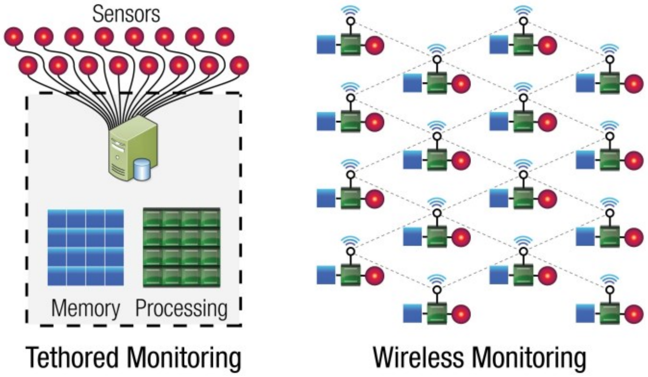


Figure 2.17 Comparison of network wide capabilities, wired versus wireless

and energy, while achieve the application objectives such as data acquisition, plant control, or system identification.

Wireless monitoring and control of civil structures has seen some great examples of application software. Automated wireless structural damage detection was first proposed by Straser and Kiremidjian, who used the Arias intensity computed at each node using local acceleration data as an indirect method to detect the energy dissipated as a component in the structure is damaged during an earthquake (Straser et al. 1998). Recent improvements on this application have focused on in network data processing, back-end cyber-infrastructure, and reliability improvements. The lessons learned from developing wireless structural monitoring systems have been extended to wireless feedback control systems. Structural control was first proposed in 1972, but has seen only limited use due to installation costs and reliability concerns (Spencer 1997). Introducing wireless control systems can remove costs associated with the installation of communication wires and increase reliability by removing a centralized point of failure. Novel developments in this application include the study of decentralization versus feedback rate (Loh et al. 2007), adaptive bandwidth algorithms for state-estimation (Swartz and Lynch 2009), and agent-based architectures that explicitly account for system nonlinearities in semi-active control systems (Kane, Lynch, and Law 2011), homotopic transformation of centralized controllers into a control distributed across sub-networks (Wang and Law 2011).

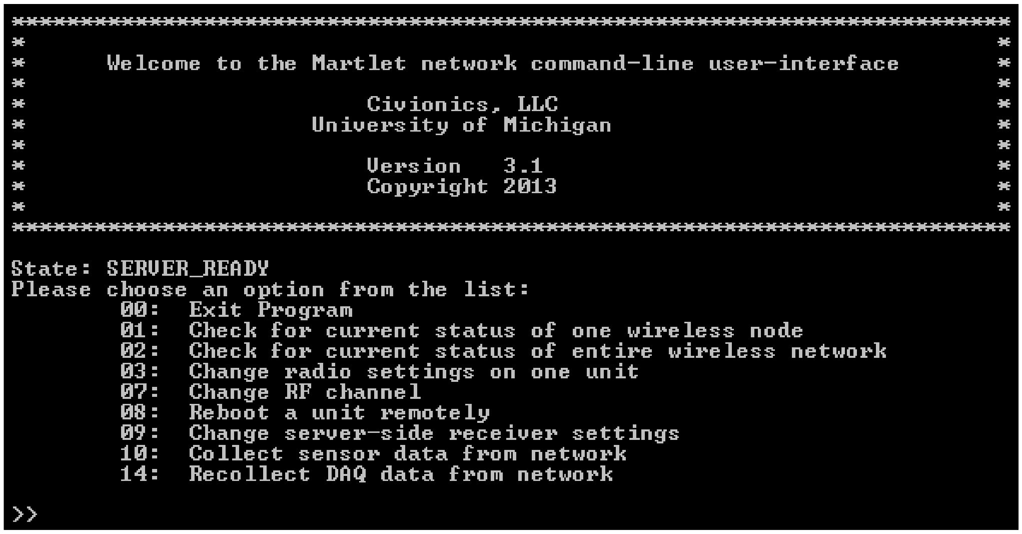
One important difference between the goals of wired CPS application software and wireless CPS application software is the need for an effective user interface (UI) for managing the wireless network and nodes. Because UI development does not directly affect the traditional CPS science, it has been largely neglected by academics. However, successful commercial implementations of wireless CPS have relied heavily well designed UI as key selling points. Innovations include an easy to use plug-and-play *WeMo* home power outlet automation system (Belkin International Inc. 2013) that can be controlled from web interfaces including the popular If This Then That (*IfTTY*) scripting website (IFTTT Inc. 2013). For home/entrepreneurial users that wish to develop their own wireless CPS, but do not want to go through effort of wireless node development *Electric Imp* has developed a WiFi based module with proprietary web service (Electric Imp Inc. 2013). Similarly the *Pinoccio* has been developed with do-it-yourself users in mind and features an easy to use web-interface, but has the advantage of a low-power IEEE 802.15.4 communication link and open-source hardware and node software (Pinoccio Inc. 2013). Focused more on the commercial and industrial side, *Device Cloud* (Etherios Inc. 2013) and ThingWorx (ThingWorx 2013) aim to create platforms that enable the Internet-of-Things (IoT). This market is quickly developing, but at the moment most of the devices are simply wireless replacements for wired monitoring and control and do not yet truly leverage the nodes embedded computing, ad-hoc communication, and distributed algorithm capabilities to truly transform the CPS paradigm toward wireless.

2.3.4 *Martlet* software development

The development of the *Martlet* required significant collaborations between all parties involved. The team took advantage of a Git version control system (Hamano et al. 2013) to manage software development, leading to over 600 contributions by over 10 authors. Additionally a wiki was used to track for hardware development and for project management. *Martlet* features that lessen the burden on the application developer include the use of industry standard C97 programming language, a simple state machine, JTAG debugging with hardware breakpoints and memory inspection. The 80 MHz floating-point CPU and parallel computation capabilities supplied by CLA further reduce the work for developers by reducing the need to write optimally efficient code. Users of the *Martlet* in the field will appreciate the *Martlets* automatic detection of peripheral boards and loading of associated calibration values. Standardization of connector types, e.g. transducer voltage requirements will prevent the field user from installing the wrong transducer into an incorrect peripheral layer port.

The *Martlet's* operating system is described by the interrupt driven state machine shown in Appendix B. After the *Martlet* initializes, it sits doing nothing until an interrupt occurs from one of the interrupt sources (i.e. a packet is received, a UART command is received, a timer counts down to zero, or one of the many other interrupts in the Peripheral Interrupt Expansion (PIE) table). If the associated interrupt enable registers are set, then the interrupt will switch the CPU from its current task to the new ISR function; after which it will return to its previous task. This lightweight OS architecture was used because the state machine can rapidly ($\sim 0.1\mu s$) switch to ISRs and can be easily understood by the developer if only one or two (multi-threaded) applications are running concurrently. Additionally, the CLA on the *Martlet* allows a completely parallel execution of code, such as a control law, to execute without having to worry about being delayed or interrupted by tasks that are handled by the CPU.

The current middleware packages for the *Martlet* have been kept as simple as possible and consist of methods for clock synchronization and ad-hoc data requests. Clock synchronization, when required for control or DAQ, occurs with a single synchronization beacon packet sent from a single node (often the gateway). This method of synchronization assumes that all the nodes that need to be synchronized are within communication range of the node sending the beacon (i.e. a star configuration or single-hop ad-hoc, see

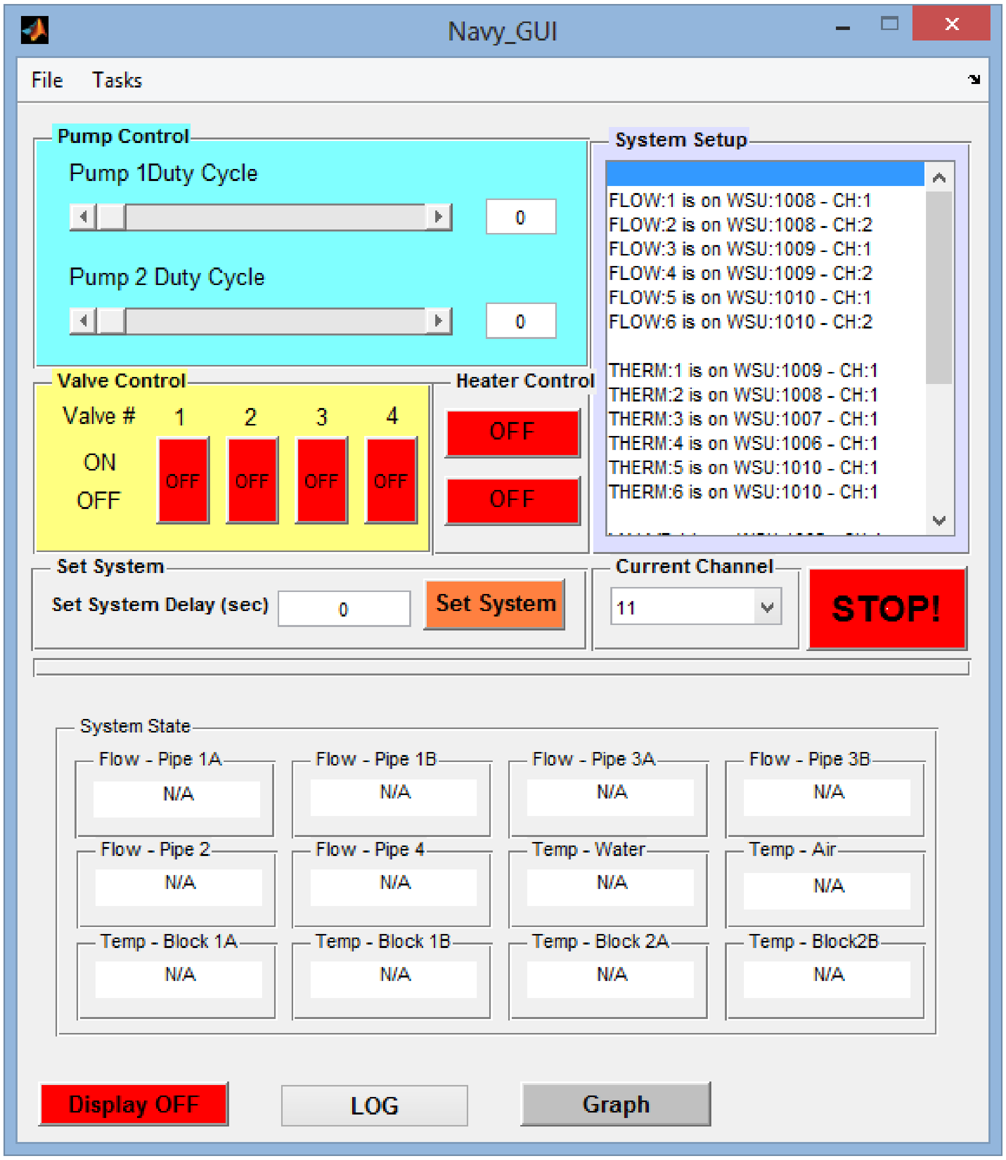


```
*************************************************************************
*                                                                       *
*      Welcome to the Martlet network command-line user-interface      *
*                                                                       *
*                         Civionics, LLC                                *
*                     University of Michigan                           *
*                                                                       *
*                         Version   3.1                                 *
*                         Copyright 2013                                *
*                                                                       *
*************************************************************************

State: SERVER_READY
Please choose an option from the list:
        00:  Exit Program
        01:  Check for current status of one wireless node
        02:  Check for current status of entire wireless network
        03:  Change radio settings on one unit
        07:  Change RF channel
        08:  Reboot a unit remotely
        09:  Change server-side receiver settings
        10:  Collect sensor data from network
        14:  Recollect DAQ data from network

>>
```

Figure 2.18 Command-line user-interface for *Martlet* networks

Figure 2.5). As such, **a cyber-physical constraint becomes apparent** in that distributed control and data processing algorithms must account for the limited ability to synchronize and communicate with all the nodes at once. For example, in Chapter 5 of this dissertation, a communication graph is created by mimicking the energy transfer graph of the physical system. When it is not possible, yet necessary, to synchronize all the *Martlets* with a single beacon the RTC *wing* (Figure 2.15) can be used to synchronize all the nodes before deployment, then rely on the accuracy of the RTC to maintain synchronization over long periods of time.

One of the key contribution of this dissertation is application software for wireless CPS. The three main applications developed are for the control algorithm for bilinear systems (Chapter 4) and agent-based distributed control of hydronic systems (Chapter 5). Chapter 6 concludes with the CPS framework in which the distribution of software application agents and their required computation tasks is affected not only by one-another, but also by the physical system that they are controlling and monitoring. These software applications are enabled by the embedding of sub-applications that were adapted from open-source PC software projects: EPANET for modeling flow in



Pump 1Duty Cycle 0

Pump 2 Duty Cycle 0

```
FLOW:1 is on WSU:1008 - CH:1
FLOW:2 is on WSU:1008 - CH:2
FLOW:3 is on WSU:1009 - CH:1
FLOW:4 is on WSU:1009 - CH:2
FLOW:5 is on WSU:1010 - CH:1
FLOW:6 is on WSU:1010 - CH:2

THERM:1 is on WSU:1009 - CH:1
THERM:2 is on WSU:1008 - CH:1
THERM:3 is on WSU:1007 - CH:1
THERM:4 is on WSU:1006 - CH:1
THERM:5 is on WSU:1010 - CH:1
THERM:6 is on WSU:1010 - CH:1
```

Set System Delay (sec) 0

Current Channel 11

Flow - Pipe 1A N/A	Flow - Pipe 1B N/A	Flow - Pipe 3A N/A	Flow - Pipe 3B N/A
Flow - Pipe 2 N/A	Flow - Pipe 4 N/A	Temp - Water N/A	Temp - Air N/A
Temp - Block 1A N/A	Temp - Block 1B N/A	Temp - Block 2A N/A	Temp - Block2B N/A

Figure 2.19 MATLAB graphical user interface for hydronic system control with *Martlets*

pipe networks (Rossman 2008), and GSL for computing analytical solutions to mathematical problems (Galassi and Theiler 2013).

The *Martlet's* ad-hoc communication and embedded computing features all for autonomous control of wireless CPS; however user interface is still required to adjust control parameters, add nodes to the network, and view results of control and monitoring. Minor changes were made to the *Narada's* command line interface to make the new UI shown in Figure 2.18 compatible with interfacing with *Martlets*, while still being backwards compatible with the

Narada. Since real-time data visualization is not built into the command line UI, the graphical user interface (GUI) shown in Figure 2.19 was developed in MATLAB that provides quick access to buttons that control pump speed and valve state. The buttons are automatically updated when the system autonomously reconfigures in addition to displaying current flow rates and temperature at each wirelessly connected sensor.

2.4 Choosing a Wireless Node: Why the *Martlet*?

Application considerations will determine which type of wireless sensor should be used in a wireless structural monitoring system. Power consumption will be less of an issue for short-term deployments of a few days while low-power consumption and power harvesting will almost certainly be required for long-term installations. The physical quantity to be measured or actuated will also effect sensor selection. While the node can be designed from scratch, it would be most advantageous to use ready-to-deploy, all-in-one commercial units if and only if available devices meet project requirements.

Even for SPM application many wireless platforms exist, and care needs to be taken to select the write platforms for the deployment and to consider the potential lack of cross-compatibility between platforms from different suppliers. Thus arises one of the challenges for the wirelessly enabled CPS community: standardization of protocols and platform interfaces will enable more straight forward adoption. More flexible wireless sensor modules, such as the 'mote' family, enable the designer to choose the desired transducer, design the power management circuitry, modify the embedded firmware, and fabricate an assembly enclosure for the specific application. If the greatest amount of flexibility is required, yet the designer wishes to not design a unit from the ground up, academic prototypes may be purchased that bring additional features such as advanced sensing (e.g., piezoelectrics (Ihler et al. 2000; Overly et al. 2008)), feedback control (Seth et al. 2005), and the greatest amount of extensibility. In 2006, Lynch and Loh (Lynch and Loh 2006) provided a review of available wireless sensors for structural monitoring at the time, but since technology advances so quickly, newer and more advanced wireless sensor models now exist. Regardless of the family of wireless sensors the user decides to choose from, it will be important that the datasheet of each device being considered is carefully studied. The following datasheet specifications are commonly misread and require intense scrutiny: active versus sleep power consumption along with the time anticipated to be spent active and sleep; maximum theoretical wireless data rate versus the realistic rate at which measurements can be reliably streamed; ADC resolution versus effective resolution including affects by circuit noise; transducer signal bounds, sensitivity, and signal conditioning circuitry; and wireless communication range in environments such as line-of-sight versus lightly or heavily constructed facilities. The remainder of this section will compare and contrasts three main families of wireless sensors currently in use (Figure 2.20) in an effort to show readers important traits that which should be broadly applicable to future generations of devices.

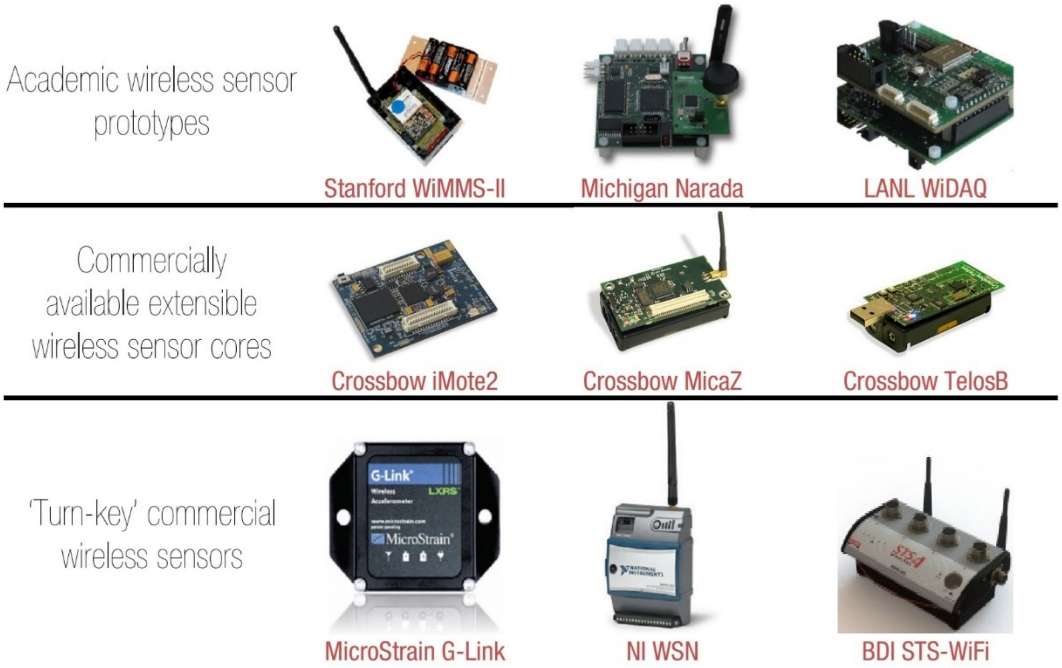

Academic wireless sensor prototypes
Stanford WiMMS-II
Michigan Narada
LANL WiDAQ
Commercially available extensible wireless sensor cores
Crossbow iMote2
Crossbow MicaZ
Crossbow TelosB
'Turn-key' commercial wireless sensors
MicroStrain G-Link
NI WSN
BDI STS-WiFi

Figure 2.20 Wireless sensor families and selected examples
(G-Link photo courtesy of LORD-MicroStrain, NI WSN photo courtesy of National Instruments and BDI STS-WiFi photo courtesy Bridge Diagnostics)

With regard to SPM, two classes of commercial wireless sensors exist: ready to deploy all-in-one units (Figure 2.7e-g) and extensible wireless sensor cores (Figure 2.7f-g). All-in-one wireless nodes such as those found in the Microstrain G/V-Link line (Microstrain Inc. 2012), Bridge Diagnostics Inc. (BDI) STS-WiFi – Wireless Structural Testing Systems (Bridge Diagnostics Inc. 2012), National Instrument (NI) WSN line (National Instruments Inc. 2012), the Ohm zSeries (Omega Engineering Inc. 2012), and WirelessHART products by companies like Siemens (Siemens AG 2012), are robust easily deployed 'turn-key' solutions to wireless monitoring. Each self-contained unit includes a power source, transducer, signal conditioning circuitry, antenna, wireless radio, and computational core. The more popular units have acquired a large user base whose collective experience can be leveraged. In order to maintain an easy to use system, many features such as maximum sampling/data rate, in network data processing, power harvesting capabilities, power management, design flexibility, and multi-hop capabilities are limited. These short comings are offset by their ease-of-use for inexperienced users and by the large user base whose collective experience can be leveraged by a novice user.

When the user has specific design requirements not met by commercial turn-key solutions, commercially available wireless sensor cores (Figure 2.7f-g) can be used to 'jump-start' development of the entire sensor node. These units typically do not include protective housings, transducers, power management circuitry; as such, peripherals must be designed or selected to bring this functionality. The 'mote' line of devices originating from Berkeley and manufactured by companies such as Intel, Crossbow, and MEMSIC provide the user more flexibility through an open source hardware and software design. The model in this line which has seen the most extensive use in the structural monitoring field is the iMote2 (Crossbow Technology 2007) , due to its embedded operating system specifically designed for WSNs, its extensibility through the use of daughter boards (e.g., the SHM-A boards (Rice and Spencer Jr. 2008) with onboard accelerometer and controllable signal conditioning), and its significant computational capabilities. In the developing field of wireless SHM, commercial units are not always available for immediate purchase to meet a project's needs. As such, the academic community and commercial sector have developed custom nodes, bringing previously unavailable features to the field.

Researchers in academia are interested in pushing the vanguard of WSN and have developed nodes with unique features not found in commercial wireless sensors at time of their development. The costs of working with cutting edge academic prototypes are the smaller user base, required programming skills, and equipment needed for assembly and debugging. The *WiMMS* sensor family developed at Stanford in the late 1990's was one of the earliest wireless sensor families for SHM. WiMMs led to the development of the *Narada* at the University of Michigan. The *Narada* was the first wireless node for civil engineering applications with wireless feedback control capabilities in the original design. Both the *WiMMS Sensor* and the *Narada* featured swappable radio modules so range, data rate, and power consumption could be tailored to the field application at hand. The wireless node developed by Bennet et al. (Bennett et al. 1999) was designed to be embedded into flexible asphalt to measure strain and temperature. Mitchell et al. (Mitchell et al. 2001) proposed a wireless node with two wireless transceivers; one to talk to low power nodes in each cluster and the other transceiver to communicate over long distances with other clusters. The WiDAQ developed at Los Alamos National Lab (Taylor et al. 2009) features a unique daughter board capable of measuring the impedance of seven piezoelectric sensors per node. Academic prototypes are similar to commercial devices, e.g. use of the IEEE 802.15.4 communication standard and 16-bit or greater ADCs. The prototype units introduced features such as feedback control, novel transducers, and dual core computations which may eventually find their way into commercial units as was seen with corrosion sensing (Inaudi and Manetti 2009).

Similar to other academic prototypes, the *Martlet* is just a platform for enabling research into wireless CPS. The *Martlet* is not a finished product ready for use by industry. One would choose the *Martlet* over one of the other nodes mentioned in this section if an extensible platform was required to build custom interfaces to cutting-edge CPS applications. The *Martlet* distinguishes itself from similar platforms with its dual core architecture ensuring precise timing of priority events, a fast processor and co-processor to execute advanced control algorithms, sturdy extensible and very customizable design (i.e. learn to develop with one platform, and it can be used for many applications), and the easy to use development environment for designers familiar with MCU development.

2.5 *MARTLET* FIELD TEST: STRUCTURAL VIBRATION MONITORING

The development process of any new wireless node should include phases of thorough testing. The *Martlet's* testing began before the first PCB was ever made. The radio was tested for reliability, strength, and sensitivity; the MCU was tested for computation speed and dual-core parallel operations; and the ADC was tested for accuracy, speed, and resolution. However, no amount of laboratory testing can validate that the device will work in real-world conditions. To this end, a programme was developed and executed as an initial field test of the *Martlet* in which its ability to be deployed as a wireless DAQ was tested on a wind turbine tower at Los Alamos National Laboratory in New Mexico, USA. Goals and motivation

Researchers at the University of Michigan, Leibniz Universität Hannover, and Los Alamos National Laboratory (LANL) were interested in investigating the long-term and transient effects of environmental operating conditions (EOC) on the structural response and performance of wind turbines. The collaborations preliminary investigation was to instrument the tower of the Whisper 500 wind turbine (Southwest Windpower Inc. 2000) installed at LANL and depicted in Figure 2.21 (located at 35°48′5"N

106°17′47.3"W) with the aim to identify key environmental and structural conditions that would affect the structural response of the tower. The structure of interest is a 12.3 m tall steel structure with a 4.5 m diameter Whisper 500 wind turbine mounted at the top. The structure was specially designed as a test bed for wind turbine research (such as that conducted by (Chipka et al. 2013)) and features a pivot point in the middle of the structure that allows the nacelle (i.e. the housing at the top of the tower) end of the tower to be lowered to the ground for maintenance purposes by unwinding a winch and removing a bolt at the base of the tower.

The need for a quickly deployable vibration sensing system capable of regularly collecting data without any supervision warranted the use of wireless sensing, making this a prime opportunity to field test an array of the newly developed *Martlet* wireless sensors. This added one more purpose to test resulting in the following three goals for the deployment:

1) Demonstrate the multi-day reliability of the *Martlet* DAQ application
2) Identify key environmental conditions that affect structural response
3) Determine other measurands besides vibrations and EOC that affect structural response.

Previous deployments of WSN on wind turbines has successfully collected meaningful structural response information with which structural modal properties were calculated (R. Swartz, Lynch, et al. 2010). Additionally, the data collected from the WSN was less noisy then the professional wired monitoring system due to noise induced by long lengths of shielded cabling. With previous tests as a benchmark, the *Martlet* should be fully capable of collecting similar response information from the LANL structure. Thus showing the *Martlet's* timing precision, ADC accuracy, and communication reliability performing as designed.

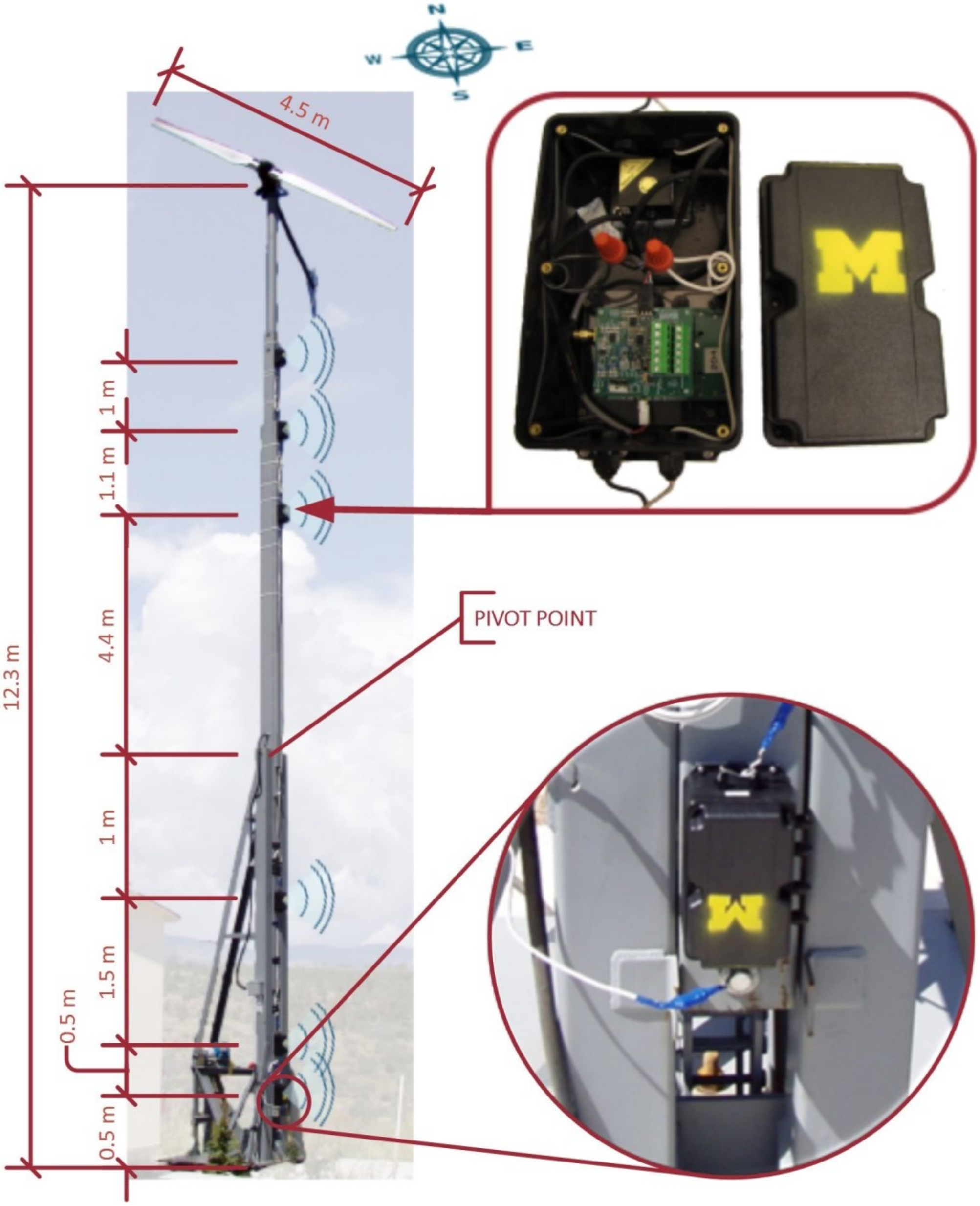


Figure 2.21 LANL Whisper 500 wind turbine and *Martlet* installation

2.5.1 Instrumentation programme

This preliminary investigation of the effect of EOC on structural response required weather data (e.g. temperature, humidity, wind speed, and wind direction) to classify the EOC and vibration data from the structural to classify the structural response. Local weather information was provided in 15 minute intervals by the LANL 'Weather Machine' (Los Alamos National Security 2013) measured at a metrological tower located 267 m from the structure at a bearing of $-73°$. In line with previous installations of WSN on wind turbines, this deployment will measure the structural response with an array of tri-axial accelerometers, specifically Crossbow CXL02TG3 high-performance accelerometers with a range of $\pm 2\ g$, low noise floor, and integral temperature sensing. A newly designed *Martlet wing*, previously shown in Figure 2.8, was used to interface the accelerometer with the *Martlet's* ADC through a 250 Hz low pass filter. Although the modal frequencies of the tower are expected to be well below 10 Hz, this high cut-off frequency was chosen in order to ensure that the high-frequency harmonics (the blades spin at a nominally rate of 500 RPM in wind speeds from 3.4 $^{m}/_{s}$ $(7.5\ mph)$ to 55 $^{m}/_{s}$ $(120\ mph)$) of the rotating blade which would be exciting the structure would be captures. The *Martlet*, Tri-ax accelerometer, signal conditioning *wing*, and power regulation *wing* were all mounted within a $21\ cm \times 12.7\ cm \times 11\ cm$ $(8.25 \times 5.00 \times 4.33")$ enclosure with $5\ V$ power cabling daisy-chaining through each box and strong mounting magnets affixed to each corner. The pivoting mechanism of the tower allowed three wireless accelerometers to be placed on the top half of the tower, and three on the bottom half of the tower. Unfortunately, the narrow section of round piping at the top of the tower prevented the magnets from securely mounting the enclosures to the top $2\ m$ $(3')$ of the structure, and the height of the pivot point prevented personnel with the available ladder from safely installing enclosures in the middle third of the structure. Although this deployment lacks a sensor at the nacelle to better capture loading and one at the pivot point to capture the boundary condition, it will still meet the requirements of the preliminary investigation described above.

The WSN was constrained by the security and safety requirements of the LANL site that prevented 24 hour access and off-site communications. Typically, previous long term WSN deployments have used a cellular modem to relay data collected from the WSN to off-site servers (Kurata et al. 2012). Instead, this server installation used a PC/104 single board computer (SBC) with a 16 GB flash drive running a Linux operating system. The large hard drive would offer plenty of room for data storage over the weeks that might have been required for the installation. Connected to the SBC via USB was an IEEE 802.15.4 compatible base-station node with external high-gain antenna that served as the gateway between the array of *Martlets* and the data collection server. The SBC and base-station were placed in an enclosure along with a power supply (PSU) connected to the 120 VAC power generated by wind turbine. The PSU supplied 5 V to the SBC and externally supplied power to all the *Martlets* on the tower. The server enclosure was placed within the shed containing all of the wind turbines power circuitry and energy storage. The external antenna and WSU power line exited the shed through an electrical conduit.

Once the sensors were deployed, the server was programmed to collected acceleration data from the X and Z axes on all units every 10 minutes at 1 kHz. Only 12 seconds of data could be collected during each 10 minute interval because of the limited RAM available on the *Martlet*. The intervals were spaced at 10 minutes to ensure that all the data could be wirelessly transmitted back to the server even in the worst case of packet drops. While personnel were at the site, the server would be manually triggered to capture interesting environmental events (e.g. sudden change in wind speed or direct) that would likely not be captured randomly within the 12 second window every 10 minutes.

Many of the data analysis methods used in SPM rely on the assumptions of white-noise or impulse-like loading to the structure. However, wind loading on turbine towers is rarely white and is typically made up of many harmonics (Holmes 2007). Therefore if only acceleration data was collected, the ability to analyze structural performance would be severely limited. This limitation is one of the key reasons that the EOC is also included in each dataset. Even with EOC and typical wind loading, it could not guaranteed that all the structural modes would be excited. To this end, the wind turbine's break could be used to generate a sine-sweep like excitation to the structure. The break is typically used to halt rotation of the blades for maintenance or safety reasons. It does not halt rotation instantaneously, but instead transforms the blades rotational energy into heat by nearly short-circuiting the output of the nacelle. This allows for the blade speed to be manually ramped-up or ramped-down in strong wind conditions, at which time data would be manually collected. With regularly collected data, plus manually collected data during interesting weather events and during manual breaking, enough quality data would be generated to meet the deployment goals outlined above.

2.5.2 Results and future outlook

Over the three days (2013-09-09 to 2013-09-09) and two nights of testing, 561 datasets were manually or automatically collected. The limited instrumentation programme of this preliminary investigation precluded the use of input-output structural response techniques since no high fidelity loading data was collected (e.g. turbine torque and RPM were not measured, and wind mean and variance data was only collected at 15 minute intervals). Data was still able to be analyzed and the structural response was still able to be extracted using output only system identification techniques.

Visual inspection of the acceleration time histories, such as those displayed for the top-most accelerometer's X- and Y-directions in Figure 2.22, show the time varying magnitude of vibration, but it is difficult to extract meaning out of the noisy-looking signals. The spectrogram plots (top Figure 2.22) of these signals, computed using a 1024 point FFT with 500 points of overlap, is more enlightening. Many strong harmonics are shown all the way up to the cut-off frequency of the anti-aliasing filter, and many of these harmonics' frequency slowly vary with time. Since blade speed was not measured, these changes in frequency of the harmonics cannot be directly accounted for, but visual observations made during testing confirm that as the blade speed decreased, the frequency of the harmonics also decreased, and *vise verse*. It is also apparent that as the blade speed, and possibly direction,

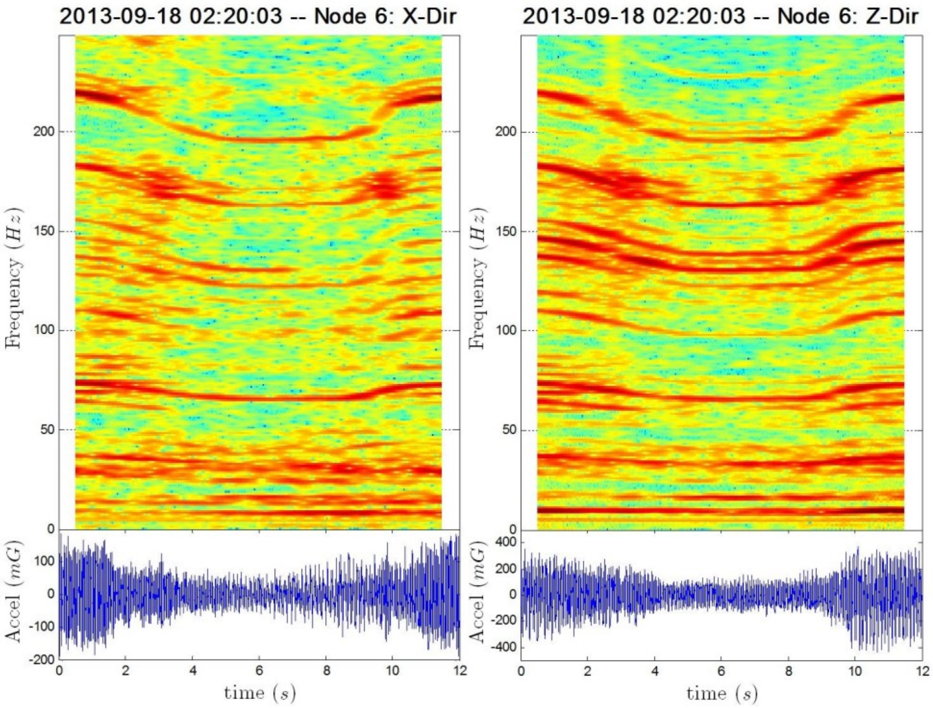


Figure 2.22Spectrogram of top-most accelerometer during 2013-09-18 02:20 dataset

change, the response amplitude of the structure changes as the excitation harmonics excite different natural frequencies of the structure. These two short signals make up only a minuscule portion of all the data collected, yet provide a surprising amount of insight. Inspecting other signals in the same way show similar insights, and some others: high frequency rattling (seen as a very broad band response) is apparent at nodes near the pivot point under some conditions,

In order to gain insight into the response of the entire structure, the signals from the X- and Z-direction of all the accelerometers was analyzed at once using the output only FDD system identification technique. Under typical civil engineering usage of the FDD algorithm, assuming a nearly white-noise structural excitation, the normal mode shapes of the structure can be extracted. The normal mode shapes can then provide useful insight into the structural performance. However, since the excitation of the tower from the turbine rotation is narrow band and full of harmonics, care must be taken to separate the resonant frequencies from the excitation frequencies, and similarly the normal mode shapes from the operational mode shapes.

The structures original design called for a normal cantilever mode shape in the E-W direction at 1.335 Hz and the other normal cantilever mode shape in the N-S direction at 1.133 Hz. Figure 2.23 shows the (normal and operational) mode shapes extracted from the acceleration data collected on 2013-09-17 at 18:45. It appears as if the order of the first two cantilever modes has switched and increased to 1.58 Hz for the N-S cantilever and 2.00 Hz for the E-W cantilever. Confidence that these are normal mode shapes and not operational mode shapes was reached after viewing the singular values of the FDD analysis done on all of the data sets collected at 10 minute intervals shown in Figure 2.24. The frequency of the peaks near 1.6 Hz and 2.0 Hz do not change significantly over time, even as the excitation does change. On the other hand, the mode shape observed at the 3.92 Hz peak in the 2013-09-07 18:45 dataset is very likely an operational defection shape. The frequency of this peak slowly drifts around 4 Hz throughout the data, and the defected shape corresponds to what would be expected from a slight imbalance in the turbine when rotating due to a wind from the south.

Observations of the spectrogram and mode shapes have been consistent with the expectation that blade rotation is the primary driver of the structural response. Comparing the spectral power of acceleration against the wind speed and direction, as shown in Figure 2.24, is then also logically consistent with the expectation that blade speed is highly correlated to wind speed. This is most obvious in comparing the movement in the peaks of spectral power between 2.5 Hz and 5.5 Hz with the variation in wind speed. At wind speeds of 5 $^{m}/_{s}$ the structure responds around 5 Hz, but as the wind speed drops to 3 $^{m}/_{s}$ the peak in response shifts closer to 3 Hz. Once the wind speed drops below 2.5 $^{m}/_{s}$ the blade rotation appears to stop causing a very significant drop in structural response amplitude. This analysis result is consistent with the Whisper 500 specification nominal blade speed of 500

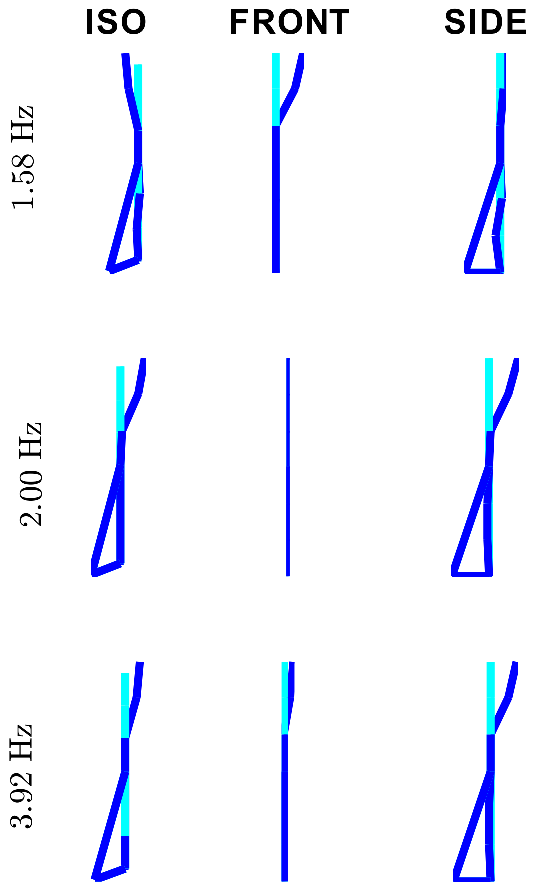


Figure 2.23Mode shapes via FDD method
Processed from 2013-09-17 18:45 dataset

RPM (8.3 Hz), cut-in wind speed of 3.4 $^{m}/_{s}$, and cut-out wind speed of 2.3 $^{m}/_{s}$. Although the excitation spectrum is not white at any given instant, considered over the entire 3 day dataset, the normal modes of the structure should be able to be separated from the operational modes. The only caveat to this desire is the very consistent wind direction. Figure 2.24 shows that the wind is almost always coming from the south with little variation (depicted as the one-standard-deviation dashed lines). This means that while the rotational vibrations in the nacelle consistently excite the E-W direction of the tower, the N-S directions might not be so well excite, thus leading to ambiguity in the modal analysis.

This preliminary investigation of both the *Martlet* and wireless SPM of high-speed wind turbines was able to achieve the original three main goals.

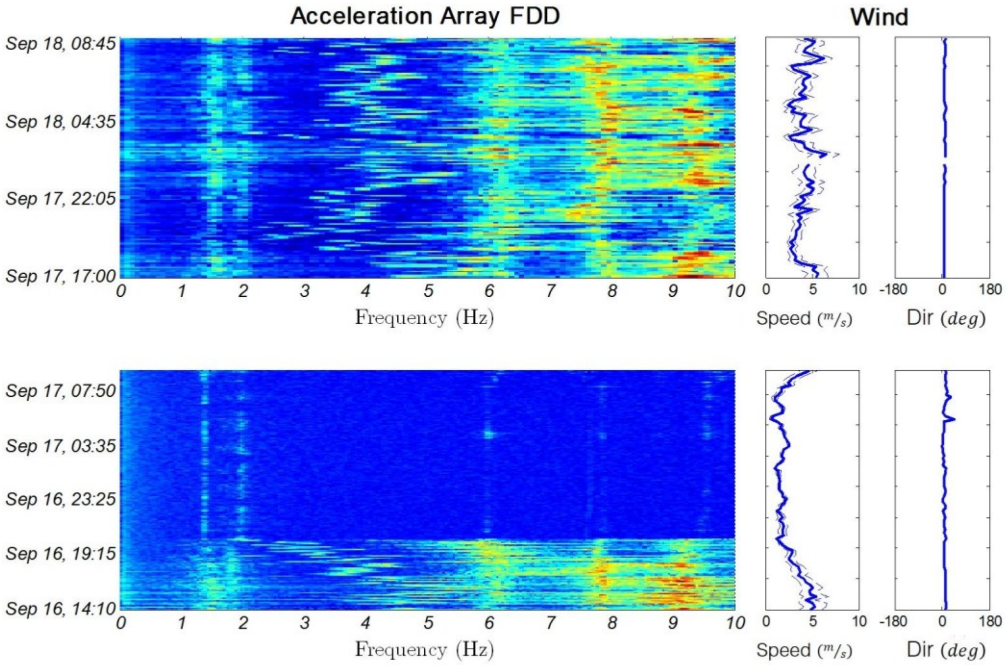


Figure 2.24Spectral power of acceleration array versus wind speed and direction

The array of *Martlets* were originally deployed on June 17th, 2013 and were not adjusted until their removal on September 18th, 2013. Severe weather prevented setup of the rest of the DAQ system until September 16th, but once setup data was successfully collected automatically and manually for three days and two nights. By analyzing the spectral power and modal response of the structural acceleration using output only system identification techniques, it was concluded that wind speed is the primary driver of structural response. The two primary cantilever normal mode shapes were able to be extracted along with the first optional deflected mode shape showing that imbalance of the rotations within the nacelle were likely the key driver to the structural excitation.

Without additional information provided by fusing data streams from other types of sensors, a more quantitative analysis of the structural response and performance is extremely difficult. Therefore, it should be proposed that further experimentation beyond this preliminary investigation should include additional sensors to try to identify the input into the structural response. A higher temporal fidelity measurement of wind speed would provide more information on the slow variations in frequency observed in Figure 2.22 spectrogram. High data-rate measurements of the instantaneous voltage and current (and corresponding phase difference) produced by the turbine would provide correlation between the instantaneous wind speed and instantaneous power generation which could be correlated to the torque applied the turbine house which then excites the structure. The existing data streams from the *Martlet* could be improved by utilizing the microSD card slot on the *Martlet* to buffer greater than 12 seconds worth of data at a time. This proposed additional information could permit the use of quantitative input-output system ID techniques which are known to provide better insight into the structural performance. It is even possible that the improved analysis could be used to identify structural damage or degradation, ultimately leading more sustainable power generation by way of reduced wind-turbine maintenance cost and greater adoption of wind energy.

2.6 THE *MARTLET* WIRELESS CONTROLLER CONCLUSIONS

This chapter presented the *Martlet* wireless device designed as an extensible platform to enable research into wirelessly enabled CPS. The design of the device's network stack was based on the OSI reference model and tailored for low power, long distance, low latency, and reliable ad-hoc communication. Additionally, the network stack is backwards compatible with other popular wireless nodes for infrastructure monitoring and control: including the *Narada*, *Formic,* and *iMote*. The hardware was designed as an extensible platform that could be applied to a wide variety of CPS applications. The dual MCU chosen as the *Martlet's* core enables control tasks to be performed with high temporal precision while less critical tasks are performed on the main core. A wide variety of digital and analog IO enable all sorts of sensors, actuators, and other peripherals to interface with the *Martlet* through custom designed *wings*, 8 of which have already been designed and built. The software design of the *Martlet* began by choosing a simple statemachine based OS that allows for rapid response of concurrent tasks in a relatively easy to understand architecture. Middleware and applications have been designed for the *Martlet* that enable its usage for sensing and control for a wide variety of application. The entire design process was culminated with a computationally capable node for distributed monitoring and control of physical infrastructure systems.

Usage of the *Martlet* entails programming applications in a high-level programming language, C, that are injected into the state-machine. Easy sending and receiving radio packets enables application writers to utilize ad-hoc communication that does not rely on a centralized network coordinator. The radio compatibility with other device designs will enable heterogeneous networks where very-low power nodes (e.g. *Formic*) can be used for remote sensing, low-power nodes (e.g. *Narada*) can be used for data aggregation and processing, medium-power nodes (e.g. *Martlet*) can be used for more advanced system ID and control, while power hungry but computational dominant nodes (e.g. *iMote*) can perform advanced computational tasks. Driving all this computation down to the wireless network, alleviates the load on centralized servers and has the potential to decrease system response and increase reliability. The key advantage to usage of the *Martlet* in these heterogeneous applications is its capability to execute timing critical complex control algorithms in a lower-power package.

In addition to controlled laboratory tests of the *Martlet*, an array of *Martlets* outfitted with tri-axial accelerometers and associated signal conditioning *wings* were deployed on a wind turbine tower at LANL. The *Martlet* was successfully leveraged to meet the project requirements of identifying the main sources of structural response and provided insight into what additional sensor fusion would be required in order to conduct system identification techniques capable of structural health and performance monitoring.

The remainder of this dissertation will present CPS approaches and algorithms that are enabled by wireless telemetry and utilize the *Martlet* platform. The development of the *Martlet* and the research presented below that it enables with open new applications for usage of the CPS paradigm with cyber components were not applicable before due to the costs and risks associated with tethered or completely decentralized SCADA systems.

Chapter 3.

CPS Paradigm Applied to Hydronic Cooling Infrastructure

This thesis applies the wireless CPS paradigm to plants included in the U.S. Navy's new integrated all-electric ship (AES) design paradigm. This application is a noteworthy example of a cyber-physical systems due to diverse physical processes, numerous hybrid control points, competing objectives, multiple tasks to be automated, and many potential points of failure on both the physical and computational side of the CPS architecture. Utilizing the wireless CPS approach to control systems design will aid systems architects in reaching the AES goal of increased automation and reliability, reduced operating costs and reduced manning requirements. For actually testing proposed wireless CPS control architectures, a laboratory test bed was created which was inspired by the hydronics, pipe network, and electrical systems onboard an AES chilled water plant. Hydronics is the term utilized for systems that transfer heat by passing water through or around an object. This test bed will serve as a proving ground for validation of the *Martlet* and the bilinear-, hybrid-, and agent-based control (ABC) architectures developed in the following chapters of this dissertation.

3.1 New frontiers in the Navy's all-electric ship

One of the branches of The United States Navy's fleet modernization efforts focuses on the paradigm shift from separate propulsion and non-propulsion electrical systems to an integrated all-electric ship (AES). In an AES, a redundant network of generators provides electric power for propulsion and all other needs (Wagner 2007). The AES concept supports the design of more efficient ships through the elimination of dual redundant power generation devices, but complicates the analysis and control of the shipboard systems through interconnection of previously uncoupled systems. For example, the chilled water network used to remove excessive heat from high-power electrical equipment will become coupled to the electrical system through the cooling of the generators and transformers and the electric actuation of the chilled water network (Srivastava et al. 2008).

Analysis and control of this added complexity will require aggregation and processing of large amounts of data. A distributed computational network may be better suited to manage this load when compared to a redundant, centralized framework that is susceptible to disproportionate damage (Zivi 2002). Existing spatially distributed sensing and control systems onboard naval vessels have employed electrical wiring or fiber optic cables for data transfer (Dunnington et al. 2003). These distributed systems can effectively and redundantly control the naval systems at the cost of expensive copper or fiber optic cables, labor intensive cable installations, and redundant bus architectures. A further reason for migrating from the wired data transfer paradigm is the Navy's drive to produce experimental littoral combat ships, *i.e.* the FSF-1 Sea Fighter (Bachman et al. 2007), aimed at high speed operation with a lightweight aluminum hull design. The additional weight of redundant data cabling becomes much more burdensome on these lightweight vessels. To this end, wireless data communication is currently being explored as an alternative by projects such as wireless integrated routing link (WIRL) (Architecture Technology Corporation 2003) and the Office of Navy Research's (ONR) reduced ship-crew virtual presence (RSVP) (Seman III et al. 2003). The University of Michigan has also explored and proved successful operation of wireless technologies for health monitoring of aluminum hull ships (R. Swartz, Zimmerman, et al. 2010) using the *Narada* wireless platform (Swartz et al. 2005).

The safety, sustainability, and performance of large-scale pipe networks, like those of a ship's chilled water system, is reliant on distributed controllers that quickly adapt to unanticipated events. As discussed above, today's pipe networks are becoming more interconnected than ever, to the point of becoming subsystems in even more complex infrastructure systems. The work in this thesis is conceptually generalizable and can be applied to any complex pipe network system. For example, the distributed controller developed for the naval test bed described in this thesis was designed for use on larger utility-scale pipe networks. Applying it to control of these systems may allow municipalities to prevent ecological disaster due to the rupture of aging pipes, extreme events, or simply reduce energy consumption.

3.1.1 Related work

Successful operation of a large-scale networked control system is critically dependent upon the communication pathways' ability to deliver information from the sensors to the controller and from the controller to the actuators in real-time. The wired communication systems used today, mainly copper wiring and fiber optics, have long been viable options for these communication systems on ships (Dunnington et al. 2003). Recent developments in wireless technology and industry standards stand to shift this paradigm. The shortcomings of the wired paradigm include the complexity and cost of installation through tight spaces, repair difficulty after inevitable damage from long-term deterioration or battle, and the rising cost of commodity copper. It has been estimated that a fifty percent reduction in cost can be seen with the adoption of a wireless communication paradigm (Seman III et al. 2003) while maintaining security with encrypted communication (Texas Instruments 2009). The difficulty of transmitting wireless data in the metallic chambers of ship hulls was shown to be a surmountable challenge in recent research for both control (Seman III et al. 2003) and ship deployed hull monitoring systems (Slaughter et al. 1997; R.

Swartz, Zimmerman, et al. 2010). These advantages of wireless control create further incentives to design such systems for shipboard control.

3.2 A TEST BED FOR WIRELESSLY ENABLED CPS

Inspired by the control systems test bed for demonstration of distributed computational intelligence applied to reconfiguring heterogeneous systems by Srivastava in (2008), a similar laboratory-scale hydronic test bed was created at the University of Michigan as part of this thesis work. This test bed served to experimentally verify the efficacy of the cyber-physical control systems proposed in later chapters of this dissertation.

3.2.1 Architecture and description

A small-scale chilled water system has been created at the University of Michigan to highlight the key functional elements of chilled water plants on naval vessels (Figure 3.1). The demonstrator incorporates four resistive heating elements bonded to the surface of four aluminum blocks to represent thermal loads the chilled water system is designed to regulate. To cool these two loads, each aluminum block (10 x 5 x 2.5 cm^3) has two cylindrical holes machined 10 cm long and with diameters of 1.25 cm. Aluminum blocks T1 and T2 in Figure 3.1 are epoxied together; while blocks T3 and T4 are paired. Two pumps are installed in the system with each pump capable of cooling the system's thermal loads through the use of automated valves and a network of pipes. These pipes are composed offlexible tubing 1.25 cm in diameter. The cooling system is divided in two major halves (*i.e.,* port and starboard halves) that are designed with a high degree of functional redundancy and interconnectedness as would be found on a ship. This allows the chilled water system to continue to meet its operational objectives even if damage was to occur on one side of the ship.

The chilled water system has two pumps denoted as P1 and P2 in Figure 3.1. The pumps selected are Greylor PQ-12 DC gear pumps capable of

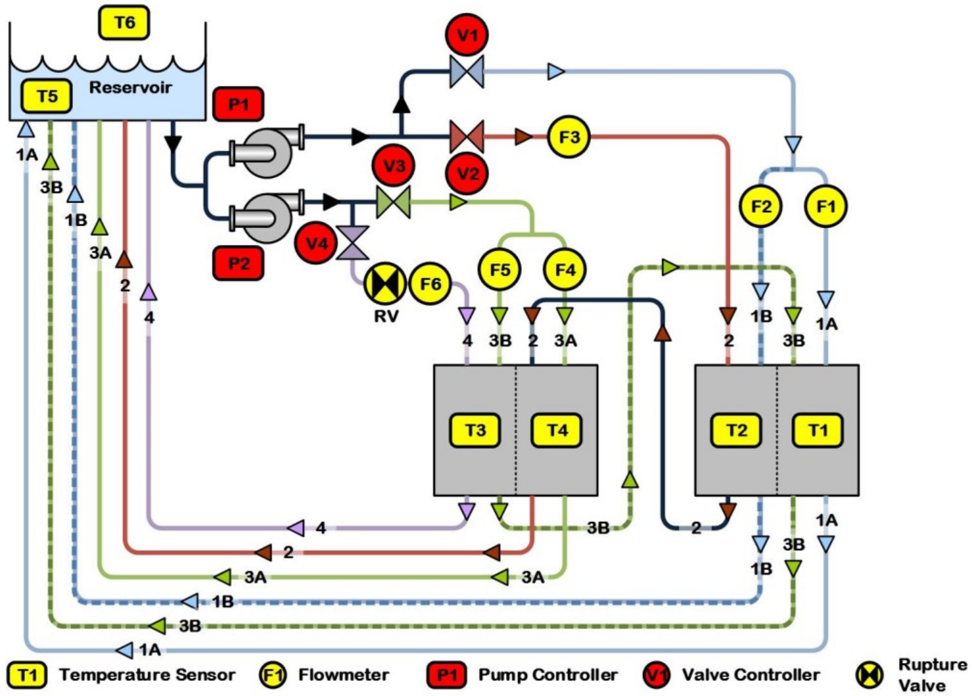


Figure 3.1 Water network schematic of U of M hydronics test bed

maximum flow rates of 37 mL/s when powered by a 12 V DC source. The output flow rate can be varied by duty-cycle application of the pump voltage source. Two *Martlet* wireless nodes are used to control the pumps with each node controlling the pump power source (a 12 V DC source) using an electrical switching circuit. Specifically, a Vishay 4N35 opto-coupler is used by each *Martlet* to duty cycle the pump using a pulse width modulation (PWM) signal generated by the *Martlet*.

The output flow of each pump is connected to two output channels controlled by valves (denoted as V1 and V2 for P1 and V3 and V4 for P2 in Figure 3.1). The normally closed STC 2W025-1/4 solenoid valves utilized to route chilled water throughout the tabletop demonstrator require a 12V supply to provide 20W of power in order to open a pipe for flow. Because the M*artlet* node itself is not capable of providing sufficient voltage and current to operate the valve, a relay is used to apply a 12 V signal from a DC power supply. This 12V power supply is electrically isolated from the *Martlet* actuation interface through the use of a Vishay 4N35 opto-coupler to protect the *Martlet*'s delicate digital and RF circuitry.

To measure the flow in the chilled water plant, six flowmeters (DigiFlow DFS-2W) are installed throughout the pipe network. The flowmeters output a digital frequency-modulated signal at 60,000 pulses per liter; this requires a high-speed sensing interface that can provide a high-resolution measurement of flow. The *Martlet*'s digital capture pins were wired to measuring the switching of the flowmeter signal and calculate the estimated flow rate.

The temperature of each aluminum block is monitored using National Semiconductor LM35DT solid-state temperature sensors interfaced directly to a *Martlet* ADC. The temperature sensors have a sensitivity of 10 mV per degree Celsius and operate between 0 and 100 °C. The four temperature sensors are denoted as T1 through T4 in Figure 3.1. Two additional temperature sensors are deployed in the system: one in the reservoir from which the chilled water is derived (denoted as T5), and the other in the vicinity of the demonstrator (denoted as T6) to measure the ambient air temperature. Figure 3.2 is a picture of the completed demonstrator.

3.2.2 Thermal system modeling and calibration

To adequately model this system an analytical model must be developed that allows the interaction between chilled water and the system heat sources to be accurately expressed. At a very basic level, the rate of heat

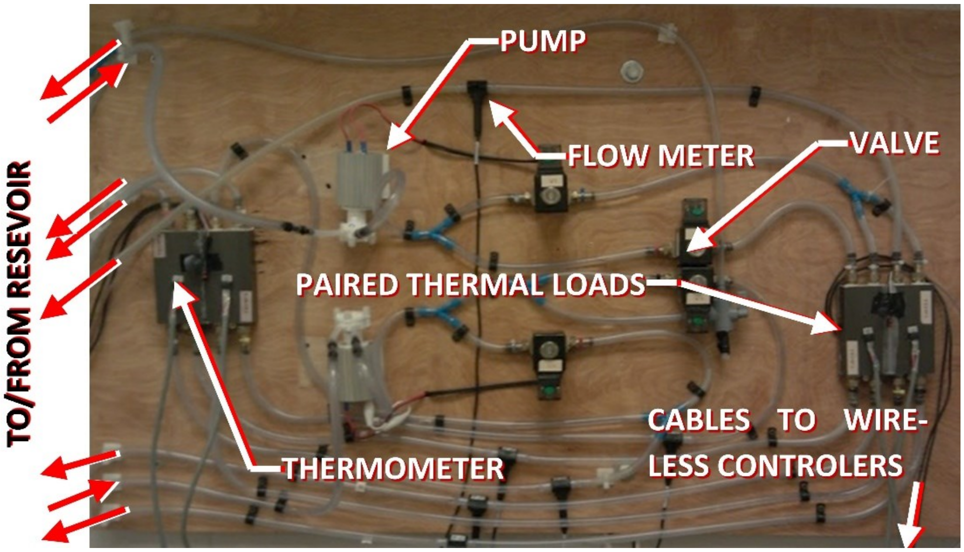


Figure 3.2 Photograph of the U of M hydronics test bed

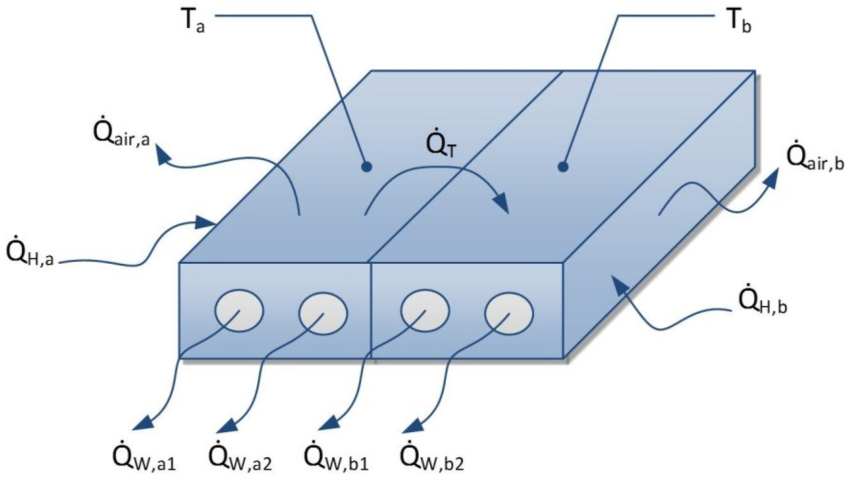


Figure 3.3 Heat-transfer flow diagram in thermal components

transfer between heat source components can be expressed as seen in Figure 3.3. From this diagram, the following basic heat transfer equations can be derived:

$$\dot{Q}_{BLOCK}^{1} = \dot{Q}_{AIR}^{1} + \dot{Q}_{WATER}^{11} + \dot{Q}_{WATER}^{12} + \dot{Q}_{BLOCK}^{12} = m \cdot c_p \cdot \dot{T}_{BLOCK}^{1} \qquad (3.1)$$

$$\dot{Q}_{BLOCK}^{2} = \dot{Q}_{AIR}^{2} + \dot{Q}_{WATER}^{21} + \dot{Q}_{WATER}^{22} - \dot{Q}_{BLOCK}^{12} = m \cdot c_p \cdot \dot{T}_{BLOCK}^{2} \qquad (3.2)$$

Using (3.1) and (3.2) as a starting point, the individual heat transfer rates for each system component can be incorporated into an equation for change in thermal energy. In this equation T is the temperature of a system component, $\dot{Q}$ is a rate of thermal energy transfer, m is the mass of one of the heat source blocks ($0.354\,kg$ for each block), and c_p is the specific heat capacity of aluminum $\left(897\ {}^{J}/_{kg\,\cdot^{\circ}\,K}\right)$. Equations (3.3)...(3.5) explore these relationships as will be described.

- For the rate of change in thermal energy from block i to the air with temperature T_{AIR}:

$$\dot{Q}_{AIR}^{i} = -\left(h_{AIR} \cdot A_{AIR}^{i}\right)\left(T_{BLOCK}^{i} - T_{AIR}\right) \qquad (3.3)$$

- For the rate of change in thermal energy from block 1 to heat block 2:

$$\dot{Q}_{BLOCK}^{12} = -\left(h_{BLOCK} \cdot A_{BLOCK}^{12}\right) \cdot \left(T_{BLOCK}^{1} - T_{BLOCK}^{2}\right) \qquad (3.4)$$

- For the rate of change in thermal energy from block i to chilled water pipe j with temperature T_{WATER}:

$$\dot{Q}_{WATER}^{ij} = -\left(h_{WATER}^{ij} \cdot A_{WATER}^{ij}\right) \cdot \left(T_{BLOCK}^{i} - T_{WATER}\right) \tag{3.5}$$

Here h is the heat transfer coefficient of a system component and A is the surface area of the component. In order to simplify future notation, the heat transfer coefficient and surface area of each component are combined as follows:

$$hA_{AIR}^{i} = h_{AIR} \cdot A_{AIR}^{i} \tag{3.6}$$

$$hA_{BLOCK} = h_{BLOCK}^{12} \cdot A_{BLOCK}^{2} \tag{3.7}$$

$$hA_{WATER}^{ij} = h_{WATER}^{ij} \cdot A_{WATER}^{ij} \tag{3.8}$$

Note that even though the heat transfer coefficient between the heat source and the chilled water should be dependent on fluid flow rate and water temperature, it is assumed for the purpose of this study that this rate is time invariant. With these modeling equation in hand, it is possible to create a state space formulation for heat transfer under different cooling conditions (e.g. air cooling, transfer between blocks, and flow through the chilled water pipes). Each separate condition is handled individually below for blocks 1 and 2. A similar analysis was completed for blocks 3 and 4.

3.2.2.1 Case 1 – Air cooling of entire block

Equations (3.1), (3.2), (3.3), and (3.4) can be combined to form the following equation for determining the heat transfer coefficient hA_{AIR}^{i} between the aluminum heat source i and the ambient air:

$$\begin{bmatrix} \dot{T}_{BLOCK}^{1} \\ \dot{T}_{BLOCK}^{2} \end{bmatrix} = \begin{bmatrix} hA_{AIR}^{1} \cdot \left(-\left(m_1 c_p\right)^{-1}\left(T_{BLOCK}^{1} - T_{AIR}\right)\right) \\ hA_{AIR}^{2} \cdot \left(-\left(m_2 c_p\right)^{-1}\left(T_{BLOCK}^{2} - T_{AIR}\right)\right) \end{bmatrix} \tag{3.9}$$

In order to empirically derive these heat transfer coefficients using data collected from the tabletop demonstrator, an experiment was carried out

Table 3.1
Heat Transfer Coefficients for Thermal System

Coefficient				
hA_{AIR}^{1}	0.1648			
hA_{AIR}^{2}	0.1542			
hA_{BLOCK}^{12}	0.6208			
	25% Speed	50% Speed	75% Speed	100% Speed
h_{WATER}^{11}	6.1135	7.2512	8.1189	8.1125
hA_{WATER}^{12}	5.3107	6.4648	8.3769	7.8091
hA_{WATER}^{21}	5.2145	6.9193	7.9766	8.2291
hA_{WATER}^{22}	5.8796	7.3270	7.9677	8.2143

where the temperature of the heat source was raised to $326^{\circ}K$ and then allowed to cool under ambient air conditions. The result of this experiment can be seen in Figure 3.4 and are tabulated in Table 3.1. It can be seen in Figure 3.4 that the experimentally determined heat transfer coefficient allows the numerical model to predict the cooling rate of each of the heat source blocks with acceptable accuracy.

3.2.2.2 Case 2 – Heat transfer between block halves

Equations (3.1), (3.2), (3.4), and (3.7) can be combined to form (3.10) for determining the heat transfer coefficient hA_{BLOCK}^{12} between the left and right sides of the aluminum thermal load.

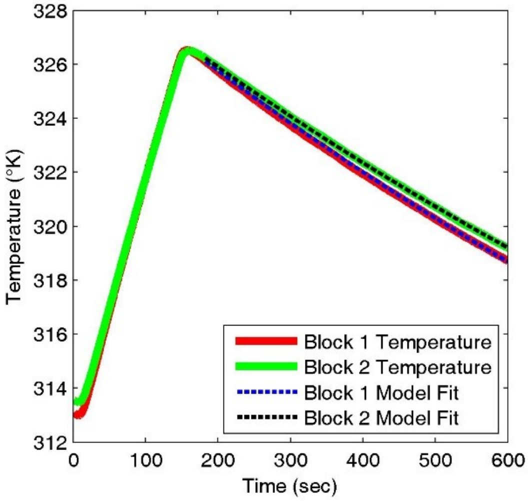


Figure 3.4 Ambient air cooling of demonstrator heat source and associated analytical heat transfer rates

$$\begin{bmatrix} \dot{T}^1_{BLOCK} + hA^1_{AIR}\left(m_1 c_p\right)^{-1}(T^1_{BLOCK} - T_{AIR}) \\ \dot{T}^2_{BLOCK} + hA^2_{AIR}\left(m_2 c_p\right)^{-1}(T^2_{BLOCK} - T_{AIR}) \end{bmatrix} = \begin{bmatrix} -hA^{12}_{BLOCK}\left(m_1 c_p\right)^{-1}(T^1_{BLOCK} - T^2_{BLOCK}) \\ -hA^{12}_{BLOCK}\left(m_2 c_p\right)^{-1}(T^1_{BLOCK} - T^2_{BLOCK}) \end{bmatrix} \tag{3.10}$$

Using the coefficient calculated in Case 1, a similar experiment was conducted to empirically derive the coefficient associated with heat transfer from one block to the other. In this experiment, the entire heat source is raised to just over $311^{\circ}K$. Then, chilled water is run through one side of the heat source, bringing the temperature of that block down to $326^{\circ}K$ and creating a substantial heat difference between the two sides of the heat source. The chilled water flow is stopped, and data gathered over the next few minutes was used to determine a value for the heat transfer coefficient hA^{12}_{BLOCK}. The results of this experiment can be seen in Figure 3.5 and are tabulated in Table 3.1.

3.2.2.3 Case 3 – Flow through individual chilled water pipes

Equations (3.1), (3.2), (3.5), and (3.8) can be combined to form (3.11) for determining the heat transfer coefficient hA^{ij}_{WATER} between block i and chilled water pipe j.

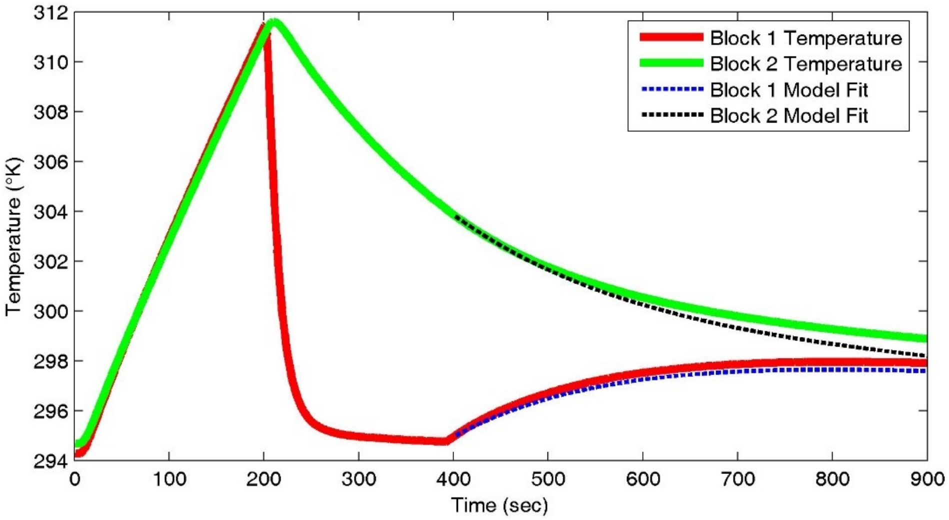


Figure 3.5 Transfer of heat between heat source blocks 1 and 2 and the associated analytical heat transfer rates

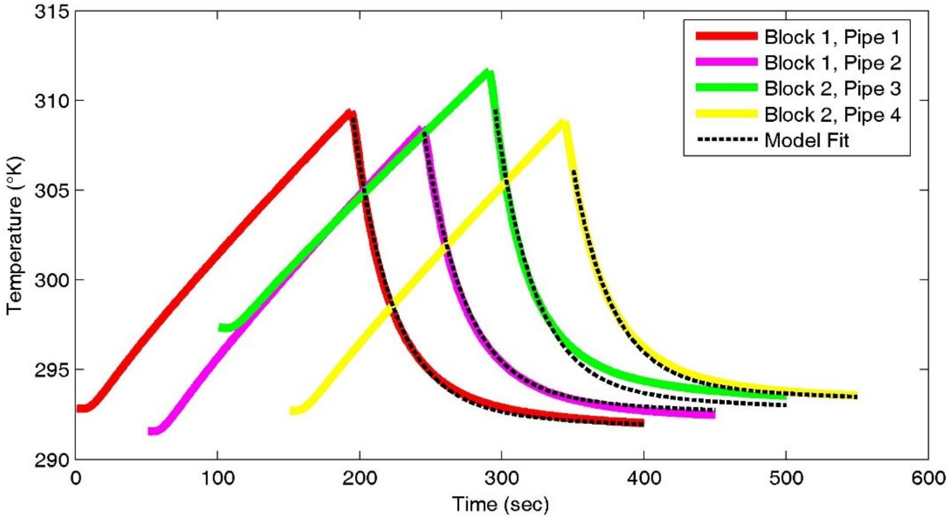


Figure 3.6 Ambient air cooling of demonstrator heat source and associated analytical heat transfer rates

$$\begin{bmatrix}\dot{T}^1_{BLOCK}\\ \dot{T}^2_{BLOCK}\end{bmatrix} = \left(m_1 c_p\right)^{-1}\left(\begin{bmatrix} -hA_{AIR} - hA^{1j}_{WATER} - hA^{12}_{BLOCK} & hA^{12}_{BLOCK}\\ hA^{12}_{BLOCK} & -hA_{AIR} - hA^{12}_{BLOCK}\end{bmatrix}\begin{bmatrix}T^1_{BLOCK}\\ T^2_{BLOCK}\end{bmatrix} + \begin{bmatrix} hA_{AIR} & hA^{1j}_{WATER}\\ hA_{AIR} & 0\end{bmatrix}\begin{bmatrix}T_{AIR}\\ T_{WATER}\end{bmatrix}\right) \tag{3.11}$$

Equation (3.11) is specific to blocks 1 and 2; a similar equation could be derived for blocks 3 and 4. Using the transfer coefficients calculated in Case 1 and 2, an experiment was conducted to empirically derive the coefficient associated with heat transfer from each block to the chilled water flowing through each of the four pipes. In this experiment, the entire heat source was repeatedly raised to just over $311^{\circ}K$. Chilled water was then run through one of the four cooling pipes. Using data gathered over a few minutes of cooling, a value for the heat transfer coefficient hA^{ij}_{WATER} was determined. The results of this experiment can be seen in Figure 3.6 and tabulated in Table 3.1.

The goal of controlling the block temperatures by adjusting pump speeds can be remapped into a constrained problem of adjusting the flow into the conduits of each block, which can further be remapped into a constrained problem of adjusting heat transfer coefficients between the blocks and water. The speed to flow map will be discussed in section 3.2.3, but the flow to heat transfer coefficient map will be discussed here. The Dittus-Boelter correlation was distilled down to the final form (3.12), which shows the relationship between the heat transfer rate and the flow rate.

$$\begin{aligned} hA_{WATER}(t) &= \frac{k_w}{D_H} N_u(t) \\ &= \frac{k_w}{D_H}(0.023)\big(Re(t)\big)^{0.8} Pr^{0.4} \\ &= \frac{k_w}{D_H}(0.023)\left(\frac{q(t)D_H}{\nu A}\right)^{0.8}\left(\frac{\nu}{\alpha}\right)^{0.4} \\ &= \frac{k_w}{D_H}(0.023)\left(\frac{D_H}{\nu A}\right)^{0.8}\left(\frac{\nu}{\alpha}\right)^{0.4}\big(q(t)\big)^{0.8} \\ &\therefore hA_w(t) = \alpha_0\big(q(t)\big)^{\alpha_1}, \alpha_0 \in \mathbb{R}^+, \alpha_1 \in (0,1] \end{aligned} \tag{3.12}$$

In this distillation the thermal conductivity k_w, hydraulic diameter D_H, Nusselt number N_u, Prandtl number Pr, Reynolds number Re, water viscosity ν, water density ρ, and water thermal diffusivity α were simplified and replaced with parameters α_0 and α_1. The monotonically increasing function $hA_w(t)$ maps $q \in [0,\infty) \mapsto hA_w \in [0,\infty)$ one-to-one and onto. Therefore this function's inverse exists, and the problem of choosing a flow rate can be algebraically abstracted to the problem of choosing a heat transfer coefficient or vice versa. The final exponential form of (3.12) has two coefficients α_0 and α_1 that are fitted to the experimentally estimated heat transfer coefficients in Table 3.1 for different flow rates. With these coefficients an accurate model of the thermal dynamics can be constructed. This model was used both for numerical simulation and for controller synthesis and design.

3.2.3 Hydraulic system modeling and calibration

The other side of the physical plant that needs to be modeled is the hydraulic system. A purely numerical hydraulic flow model, such as those that utilize EPANET (Rossman 2008), could have been created for simulation purposes, but the model would have required experimental validation and the limited size of the test bed permitted a purely empirical approach. As such, a comprehensive experiment was conducted for each of the possible valve configurations, in which the speed of each of the pumps was ramped up in 10% increments and the flow in each of the six pipes was measured along with the average power consumed by the pumps. Once the data was collected, polynomial curves were fitted through the data to generate maps from pump speed (i.e. duty cycle) to pipe flow, flow to speed, speed to power, and flow to power. A subset of the experimental data and fitted curves, for the case with all valves open, is plotted in Figure 3.7 and shows a well fitted model.

Using the models of the hydronics and hydraulics from sections 3.2.2 and 3.2.3 respectively, a complete computer simulation environment of the environment was implemented in MATLAB using numerical time-history integration techniques. This model was also used for the model based controllers presented later in this dissertation.

3.2.4 Proposed test scenarios

In order to benchmark future controllers' operation on the demonstrator, test scenarios must be developed that switch on the heaters at given times while the controller is running in order to excite as many of the system 'modes' as possible. These scenarios should be typical of what a controller would experience onboard ship and demonstrate the controllers' performance in steady-state and transient environments. The method of empirical validation of new advanced controllers has already been applied to two agent-based controllers (Kane and Lynch 2012; Kane, Lynch, and Zimmerman 2011) and will be applied to the controllers developed as part of this thesis.

3.3 The potential for wirelessly enabled CPS in the AES

In this chapter the United States Navy's new all-electric approach to ship design was explored from the point-of-view of of cyber-physical systems. Motivated by this potential application, a laboratory bench-scale test bed was

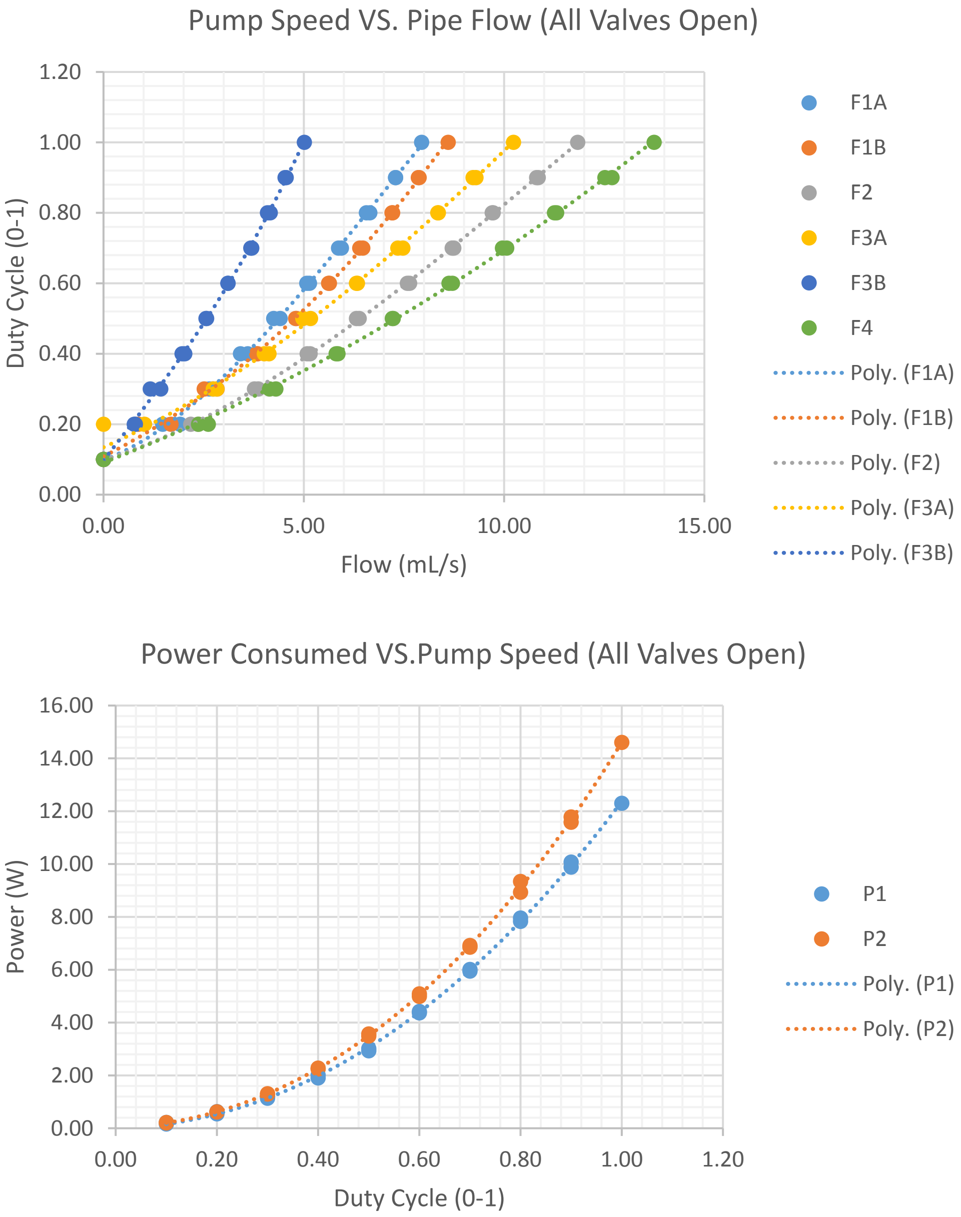


Figure 3.7 Hydraulic network model fitting

designed, assembled, and modeled for the purpose of benchmarking the performance of new controllers. The test bed exhibits characteristics common to many cyber-physical infrastructure systems, which enables conclusions to be made about controller performance on the test bed that maybe abstracted to other application areas. This benchmarking approach will also be utilized by both Chapter 4 and Chapter 5 for testing the innovative controllers of this thesis.

Chapter 4.

Bilinear Systems (BLS): Continuous Dynamics and Control

Bilinear systems (BLS), a special class of nonlinear dynamic systems, are studied in this chapter. The study is motivated by the need to control hydronic cooling systems such as those found on naval vessels. Additionally, the contributions made herein aim to be generalizable to other applications that exhibit bilinear behavior such as semi-active vibration control and control of building HVAC systems. This chapter's developments are applied to a system mathematically similar to the hydronic systems on AESs (see Chapter 3) but simpler, for the sake of easing exploratory analysis. The goal of the controllers under consideration is to maintain thermal load temperatures within a safe region while expending minimal energy pumping chilled water through the loads. The chapter begins by introducing BLS, against the traditional context of linear state-space control systems. With the foundation laid, the chapter progresses onto a discussion of open-loop (OL) control of hydronic systems using the BLS framework in which objective functions are defined and solutions to the optimization problem are offered. Section 4.4 proposes to increase the robustness of the OL controllers by incorporating them into a model-predictive control (MPC) architecture. Along with the proposed architecture, computer algorithms are presented to execute the MPC on the *Martlet* wireless controller. The BLS MPC proposals made throughout the chapter are validated in Section 4.5 on a single *Martlet* controlling a subsection of the hydronics test bed. The chapter culminates with a summary of the contributions and a discussion on the potential for the contributions to enable more viable distributed control solutions to wirelessly enabled cyber-physical infrastructure.

4.1 INTRODUCTION TO BLS

The theory of controlling CPS borrows a tremendous amount of knowledge from the traditional control system engineering community. When possible, control system engineers prefer to model a plant as a linear time-invariant (LTI) first-order dynamic system, since the theory is well established and based mostly on computationally efficient linear algebra. LTI systems can be written in the mathematical form of (4.1) in which the n instantaneous states of the system are grouped and defined as the vector $x(t) \in \mathbb{R}^n$, and the m control inputs are defined as the vector $u(t) \in \mathbb{R}^m$ with matrices A and B of appropriate size. In addition to this convenient form, LTI systems feature other desirable properties, such as the separation principle (Brezinski 2002) that allows controllers and observers to be designed independently. None of these convenient properties inherently lend themselves generally to non-LTI systems. Unfortunately, not all systems can be modeled as LTI, and often those that can be modeled as LTI become nonlinear once limitations on the control authority are considered. As such, it is often helpful to identify specific classes of nonlinear system systems that exhibit special properties or forms that make their control design easier. Bilinear systems[1] are one such class in which the state dynamics are mathematically described by (4.2) in which each scalar input $u_i(t)$ is multiplied by the state $x(t)$ through a vector B_i in addition to the $b_i u_i(t)$ term found in LTI systems. Without control, the system evolves autonomously and linearly according to the drift term $Ax(t)$.

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{4.1}$$

$$\dot{x}(t) = Ax(t) + a + \sum_{1}^{m} u_i(t)(B_i x + b_i) \tag{4.2}$$

[1] The form of (4.2) is commonly called ‘bilinear’; however some (e.g. Sontag, Tarn, et al.) define (4.2) as ‘biaffine’ and reserve ‘bilinear’ for systems in which every b_i is zero which are otherwise called homogeneous bilinear systems (Elliott 2009).

Bilinear system theory was first introduced in the U.S. by Mohler in 1966 (Elliott 2009). Since then, a thorough but not yet complete theory of BLS has been developed that provides optimal controls, stability guarantees, and even controller-observer separation principles to certain sub-classes of BLS. These are leveraged to the fullest extent possible in this chapter. However, that extent is limited due the scope of most of the aforementioned developments not including either bounded controls, drift terms, or disturbances found in the plants of interest to this chapter.

The development of the BLS theory has been motivated by its applicability to a wide variety of natural, biological, and man-made processes. In problems associated with the control of a population x, the birthrate minus death-rate u can be controlled leading to the simplest BLS in the form of $\dot{x} = ux$ (Mohler 1970). Similarly, the catalyst concentration in chemical reactions and the enzyme concentration in biochemical reactions can be modeled as control inputs of a BLS (Mohler 1970). The thermal-regulation in warm blooded animals can be modeled as a BLS in which the conductance between the skin and air can be adjusted via perspiration and the conductance between the core and skin can be adjusted via vasomotor control of circulation (Mohler 1970). Outside the biological realm, physical systems such as regulation of thermonuclear reactions and automobile braking can also be modeled as a bilinear control system.

In the field of civil infrastructure engineering, BLS are used to model and design controls for a wide variety of applications. Traffic flow dynamics on freeways are modeled using the Cell Transmission Model (CTM). This model can be formulated as a stochastic discrete time BLS in which traffic density evolves in a bilinear manner with respect to the stochastic mode of operation, e.g. free-flow, congestions, and congestion waves (Zhong and Sumalee 2008). Structural vibration control using semi-active actuators such as magnetorheological (MR) dampers is a nonlinear control problem that has been formulated in a variety of ways (Spencer and Nagarajaiah 2003), and although the BLS approach was not used originally it is gaining traction in the community. Scruggs, et al. points out the differences between BLS and linear time-varying (LTV) models of semi-active control. The BLS model provides the advantage of instantaneous, time-invariant, algebraic constraints imposed on the control inputs as opposed to the time-varying state-dependent constraints found in the LTV formulation (Scruggs et al. 2007). Even active vibration control systems exhibit bilinear behavior when the control authority is limited by physics. This phenomena is seen in the vibration control of guy cables by axial displacement of supports (Susumpow and Fujino 1995). Similar to structural vibration control, but outside of the field of civil infrastructure engineering, the control of automobile vibrations has leveraged the BLS approach (Elbeheiry et al. 1995). Recent interests in reducing energy consumption by building mechanical systems has attracted experts in BLS to the problem yielding promising results (Kelman and Borrelli 2011; Naidu and Rieger 2011).

Similar to the plant studied in this chapter, BLS have been used to model control systems for many different types of thermal processes. A BLS formulation can be used to enable self-tuning control (STC) of high-temperature furnaces used for heat treatment that can be represented as a single zone model of well mixed gasses radiating heat to the load. The STC, an adaptive control scheme, consists of a parameter estimator and a control law implemented to improve upon non-adaptive control. The improvement is due to its ability to re-tune the controller as the plant changes as a result of to slow nonlinearities. MPC can be used to realize the STC through a quasi-linearization of the BLS which utilizes the STC to adapt to changes in the quasi-linearization (Burnham et al. 1994). For the linear state-space control of steam super-heaters, an observer is necessary to estimate the state (i.e. temperature) of the metal plate separating the flue gas from the super-heated steam. The observer can be formulated with the metal temperature regarded as an unknown input into a BLS modeling the evolution of steam temperatures. Using a known polynomial trend of metal temperatures along the super heater along with the bilinear observer (analogous to a bilinear controller), a linear controller is able to outperform PID control systems (Lee et al. 1997). Similarly, BLSs can be used to model and control cooling systems such as building campus chiller plants with thermal energy storage. In one such implementation, the MPC algorithm uses a mixed-integer non-linear program (MINLP). The input to the BLS is the switching signal to turn the chiller on or off, and a heuristic is used to ease the computationally difficult problem of solving the MINLP online without a significant loss in performance (Deng et al. 2013). This chapter will continue to develop on the same track as these works and others to realize a controller for the hydronic system described in Chapter 3.

Controller designs for BLSs are influenced by a variety of factors including the objective, specialized forms of BLSs, and control hardware. The largest body of BLS controls research is on system stabilization. That is, the study of determining either an OL control trajectory or CL feedback function that ensures the system remains within a finite bound from an equilibrium position. The most common method of developing stabilizing control is by Lyapunov's Direct Method in which nested smooth hypersurfaces are found from which the system cannot leave once they have been entered. The drawback of this method is that no guarantees on optimality are inherent to the controller design process. To this end, optimal control theory was developed which ensures that a predetermined objective function is minimized by the specified control. The most studied optimal control was time-to-target with piece-wise constant (PWC) controls. Unlike linear systems, where optimal feedback functions can be found a priori by analytically solving the Algebraic Riccati Equation, determining optimal feedback functions for BLS can be a significant computational burden. However, by assuming that actuation is PWC, which occurs with digital controllers, controls can be computed as OL switching times, or as CL hypersurfaces on which controls should switch (Elliott 2007). Care must be taken when designing and simulating such switching controllers to prevent zeno behavior, i.e. the control switches an infinite number of times within a finite time period.

Analogous to the use of linear algebra for the analysis and design of linear control systems, Lie algebra serves as a tool for analyzing and designing bilinear control systems. For example, the controllability of homogenous BLS is related to the Lie algebra of the matrices A and B, similar to the Kalman rank condition for controllability of LTI systems. In essence, the matrix Lie (pronounced 'lee') algebra used in BLS is a special linear subspace of $\mathcal{M}^n$ (i.e. the linear space of all $n \times n$ matricies) that is closed under the Lie bracket $[X, Y] = XY - YX$. See (Belinfante and Kolman 1989) for a survey of facts, applications, and methods for Lie algebra. Due to the complexity and mathematical restrictions on developing CL control for BLS using Lie algebra, other methods have been borrowed from nonlinear control theory. Besides Lyapunov controllers previously mentioned, clipped-optimal control and model-predictive control (MPC) has seen extensive use in the control of BLS. Clipped-optimal control is a sub-optimal controller that reformulates a BLS with rectangular bounds on control into a LTI system with state dependent bounds on control, then implements a linear optimal control policy and 'clips' the magnitude of the control if it is outside the bounds. On the other hand, MPC directly accounts for the BLS and its bounds by optimizing the OL control trajectory over a finite prediction horizon, then iterates this OL trajectory optimization at fixed intervals in time in order to adjust to changes or errors in the model, disturbance, or reference. The controller developed in the remainder of this chapter will utilize the MPC architecture for thermoregulation of a hydronic cooling system.

4.2 The chilled-water plant as a BLS

Hydronic systems use water, or a similar fluid, as a media for removing, transferring, or adding, thermal energy. In typical two-pipe hydronic cooling systems chilled water is pumped from a reservoir through the 'chilled water supply' pipe network which delivers the chilled water to the thermal loads. In the process of passing through the thermal loads' heat exchangers, the chilled water heats up and continues to flow through the 'chilled water return' back to the reservoir, possibly passing through a chiller beforehand. Hydronic systems are preferred over other methods of heat exchange due to the high thermal capacity of water and the ease of directing the energy through a pipe network. Because of these advantages, hydronic heating and cooling systems are found in many industries including building energy systems, process control, and onboard naval ships.

The hydronic systems found onboard naval ships are part of the highly integrated and coupled ship system that includes electric power, propulsion, high-energy weapon systems, and various auxiliary systems (Zivi 2002). The hydronics are coupled mainly with the electrical power used to run the pipe network's pumps and valves, and with the heat transfer between the cooling fluid and the ships heat generating components. Due to the complex nature of this coupling, a centralized control architecture is not ideal. Instead a distributed control [1] architecture is desired that is fault-tolerant with maximum control authority pushed to the lower levels of the control system hierarchy (Srivastava et al. 2008).

Due to the complexities of even the bench-scale hydronics test bed presented in Chapter 2, and the interest in initial developments showing that a MPC can be applied to hydronic systems and embedded into the *Martlet*, a simpler hydronic system was analyzed. This simple system, shown in Figure 4.1, includes a single pump delivering chilled water from a reservoir to a single

[1] Distributed control implies that the subsystem controllers communicate with one another, while decentralized implies complete decoupling.

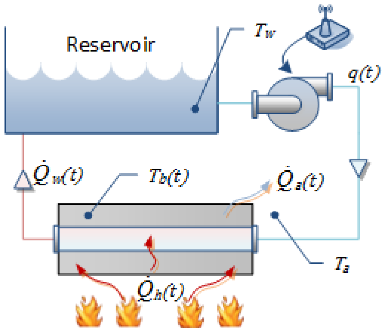


Figure 4.1 Schematic of simplified plant.

thermal load which is heated from a resistive heater and also cooled through natural convection with the air. A single *Martlet* interfaces with the temperature sensor on the thermal load and the pump motor controller.

The temperature T_b of the thermal load with mass m and specific heat capacity c_p is a function of the energy transfer shown in Figure 4.1 and modeled by (4.3). Thermal energy is gained from the resistive heater, $\dot{Q}_h$; lost through natural convection, $\dot{Q}_a$, to the air with constant temperature T_a; and lost through forced convection, $\dot{Q}_w$, to the water with constant temperature T_w. These convection terms are modeled by (4.4) and (4.5) respectively where the parameters h_a and h_w model the heat transfer coefficient from the block to the air and water respectively.

$$mc_p\dot{T}_b(t) = \dot{Q}_w(t) + \dot{Q}_a(t) + \dot{Q}_h \tag{4.3}$$

$$\dot{Q}_a(t) = \big(T_a - T_b(t)\big)h_a \tag{4.4}$$

$$\dot{Q}_w(t) = -\big(T_w - T_b(t)\big)h_w(t) \tag{4.5}$$

The goal of controlling the block temperature dynamics is achieved by adjusting the heat transfer rate between the block and water by adjusting the flow rate $q(t)$. The Dittus-Boelter correlation (Bergman 2011) was distilled down to the final form of (4.6) as a relation between the heat transfer rate and the flow rate. In this distillation the thermal conductivity k_w, pipe surface area A, hydraulic diameter D, Nusselt number N_u, Prandtl number Pr, Reynolds number Re, water viscosity ν, water density ρ, and water thermal diffusivity α

are replaced with model parameters α_0 and α_1 determined experimentally. The monotonically increasing function $h_w(t)$ maps $q \mapsto h_w$. Therefore this function's inverse exists, and the problem of choosing a flow rate can be algebraically abstracted to the problem of choosing a heat transfer coefficient or vice versa.

$$
\begin{aligned}
h_w(t) &= \frac{k_w}{AD_H} N_u(t) \\
&= \frac{k_w}{AD_H}(0.023)\big(Re(t)\big)^{0.8} Pr^{0.4} \\
&= \frac{k_w}{AD_H}(0.023)\left(\frac{q(t)D_H}{\nu A}\right)^{0.8}\left(\frac{\nu}{\alpha}\right)^{0.4} \\
&= \frac{k_w}{AD_H}(0.023)\left(\frac{D_H}{\nu A}\right)^{0.8}\left(\frac{\nu}{\alpha}\right)^{0.4}\big(q(t)\big)^{0.8} \\
&= \alpha_0\big(q(t)\big)^{\alpha_1}, \alpha_0 \in \mathbb{R}^+, \alpha_1 \in (0,1]
\end{aligned}
\tag{4.6}
$$

Using (4.3)…(4.6) the dynamics of the block temperature can be expressed in the following form:

$$
\begin{aligned}
\dot{T}_b(t) = \left(-\frac{1}{mc_p}h_a\right)T_b(t) + \big(q(t)\big)^{\alpha_1}\left(-\frac{\alpha_0}{mc_p}\right)T_b(t) \\
+ \big(q(t)\big)^{\alpha_1}\left(\frac{\alpha_0}{mc_p}\right)T_w + \left(\frac{1}{mc_p}\big(h_aT_a + \dot{Q}_h\big)\right)
\end{aligned}
\tag{4.7}
$$

Further inspection of (4.7) shows that the control problem will be nonlinear because the controlled parameter $q(t)$ is multiplied by the state $T_b(t)$ instead of being added linearly. Additional nonlinearity is introduced by the constraint imposed by the finite capacity of the pump which leads to a bound on flow rate $q(t) \in [0, q_{max}]$ and thus a bound on the heat transfer coefficient to water $h_w(t) \in \left[0, h_{w_{max}}\right]$.

The system nonlinearities are due to the constraints on control and the way the control is introduced into the state equation; however, if the flow rate is held constant, the state equation boils down to a linear autonomous system with steady state response described by (4.8). It can easily be shown that if the system starts at an initial temperature T_{b_0} with a constant flow rate q, then it monotonically approaches the equilibrium temperature $T_e(q)$.

$$\lim_{t\to\infty} T_b(t) = T_e(q) = \frac{\left(q(t)\right)^{\alpha_1}\alpha_0 T_w + h_a T_a + \dot{Q}_h}{h_a + \left(q(t)\right)^{\alpha_1}\alpha_0} \tag{4.8}$$

4.3 Open-loop optimization of BLS

Given these physical relations that describe the system, a controller can be designed such that the system performs as desired. But how does that controller achieve the desired response? And what, specifically, is the desired response? Is there more than one way of achieving such performance?

In this section, the focus will be on determining a desirable schedule of controls $\boldsymbol{u} \coloneqq \{u(t) : t \in [0, t_f]\}$, i.e. an open-loop (OL) control trajectories, that produce the desired evolution of the system's state. This is in contrast to a closed-loop (CL) controller, i.e. feedback controller, that uses measurements of the state to correct any errors in the control signal due to system noise or model inaccuracies. Thus far, the exact meaning of the control variable $u(t)$ has not been defined. This is because the choice of the physical meaning of $u(t)$ affects the form of the mathematical model that describes the plant evolution. Section 4.3.1 will discuss two such possible model formulations: a linear plant with state-dependent constraints on control, and a BLS with rectangular control constraints. With a model in hand, Section 4.3.2 develops three objective functions that could be used to design an optimal control schedule that minimizes the desired objective. Finally, Section 4.3.3 proposes realizations of the controllers developed in 4.3.2 by presenting algorithms to solve for the minimizing control policy.

4.3.1 Bilinear versus linear models

Although the control problem is nonlinear, the system can be formulated as a linear system in which the controlled parameter $u(t)$ captures

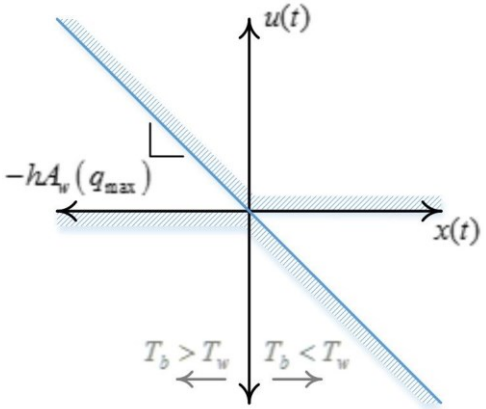


Figure 4.2 Linear model state-dependent control constraints

the nonlinearity in a state dependent constraint (4.10). The state equation of $x(t)$, a function (4.9) of T_b and the set point T_{set}, can be expressed as (4.11). The state transformation matrix A_L, the control transformation matrix B_L, and the disturbance G_L are described by (4.12), (4.13), and (4.14) respectively.

$$x_L(t) = T_b(t) - T_{set} \tag{4.9}$$

$$u(t) = -(x(t) - T_w + T_{set})h_w(t) \tag{4.10}$$

$$\dot{x}_L(t) = A_L\, x(t) + B_L\, u_L(t) + G_L \tag{4.11}$$

$$A_L \coloneqq -\left(mc_p\right)^{-1} h_a < 0 \tag{4.12}$$

$$B_L \coloneqq \left(mc_p\right)^{-1} > 0 \tag{4.13}$$

$$G_L \coloneqq \left(mc_p\right)^{-1} \left((T_a - T_{set})h_a + \dot{Q}_h\right) \tag{4.14}$$

$$\mathcal{U}_L(x_L, u_L; t) = \left\{x_L, u_L \colon \left(u_L + (x_L - T_w + T_{set})h_{w_{max}}\right)u_L \leq 0\right\} \tag{4.15}$$

The control parameter $u_L(t) = \dot{Q}_w(t)$ is constrained by the nonlinear state dependent constraint depicted in Figure 4.2 and expressed as (4.15). For a given air temperature, water temperature, and heat disturbance, mapping from u_L to the heat transfer coefficient h_w is a matter of an algebraic relationship that is one-to-one and on-to over the admissible controls for non-zero states.

System models of BLSs contain a term that multiplies the state by the control input, resulting in a LTI autonomous system when constant control is used, but LTV when the control is varying. Modeling the simplified hydronics plant in this manner by defining the control as the heat transfer coefficient, $u_{BL}(t) = h_w(t)$ yields the state equation (4.16) with parameters (4.17)…(4.20).

$$\dot{x}_{BL}(t) = A_{BL}x_{BL}(t) + (B_{BL}x_{BL}(t) + b_{BL})u_{BL}(t) + G_{BL} \tag{4.16}$$

$$x_{BL}(t) \coloneqq T_b(t) - T_{set} \tag{4.17}$$

$$A_{BL} \coloneqq -\left(mc_p\right)^{-1}h_a < 0; \quad B_{BL} \coloneqq -\left(mc_p\right)^{-1} < 0 \tag{4.18}$$

$$b_{BL} \coloneqq \left(mc_p\right)^{-1}(T_w - T_{set}) \tag{4.19}$$

$$G_{BL} \coloneqq \left(mc_p\right)^{-1}\left((T_a - T_{set})h_a + \dot{Q}_h\right) \tag{4.20}$$

$$0 \le u_{BL}(t) \le u_{BL_{max}} = h_{w_{max}} = \alpha_0 q_{max}^{\alpha_1} \Rightarrow \mathcal{U}_{BL} = [0, h_{max}] \tag{4.21}$$

The key advantage of using the bilinear model is the simplification of the control constraints from nonlinear and state-dependent to a time-invariant rectangular constraints (4.21). Once the control $u(t)$ is determined, the mapping (4.6) can be used to allocate the appropriate flow of chilled water to the thermal load.

4.3.2 Objective functions

In general, the goal of the controller is to maintain the block temperature within a safe operating range while minimizing the amount of energy consumed by the pump. This general objective can be defined in a variety of exact and approximate ways outlined in the subsections below. Common to all these formulations is the assumption that the external disturbances T_a, T_w, and $\dot{Q}_h$ and the desired temperature are held constant, and that the controlled flow must remain in the bound $q(t) \in [0, q_{max}]$ over the time window.

The minimum-time objective aims to drive the block temperature to within the safe region as quickly as possible, with no penalty added for exercising control. Analytically this is described by (4.22) in which t_f is the time at which the block temperature $T(t)$ is within the safe temperature set $\mathcal{T}_{set} = [0, T_{set}]$, such that the flow produced is feasible for all time. If the system begins within $\mathcal{T}_{set}$, then no control is prescribed by the minimum time objective, but another control could be used to try to maintain $T \in \mathcal{T}_{set}$ (e.g. the minimum-power thermal-limit protection objective). Realization of a controller with this objective must include an analysis of system parameters and initial conditions for which it is possible to drive the temperature within the safe region in finite time. If it is plausible that a situation could occur where $t_f \to \infty$, an alternate control strategy shall be deployed.

$$\begin{gathered}\min_{t_f, q(t)} t_f = \min_{t_f, q(t)} \int_0^{t_f} dt \\ \text{s. t.} \left(T\left(t_f\right) \in \mathcal{T}_{set}\right); \left(T(t) \notin \mathcal{T}_{set}\,; q(t) \in \mathcal{q} = [0, q_{max}]\ \forall\, t \in \left[0, t_f\right)\right)\end{gathered} \tag{4.22}$$

The quadratic cost function (4.23) minimizes the time-integral over the fixed horizon $t = 0 \ldots t_f$ of both the deviation from the desired temperature T_{set} and the amount of flow required. This objective goes beyond the general objective outlined above since error is penalized when the temperature is both above and below the safe set-point temperature. This heuristic is justified by the desire to minimize the wasted energy used when the temperature is driven further below the set-point. The additional cost term Ru^2, a quadratic cost on control u as defined by the problem formulation[1], is added in order to ensure a mathematically well-posed optimization problem and to minimize the energy expended by the pump. This heuristic based approach with a common quadratic cost form may lose attractiveness for more complex systems where the attribution of cost associated with excess flow to a specific pipe may be undefined. Also, the quadratic cost associated with the violation of the safe

[1] It should be noted that the control term u used in this objective function is a dependent upon the model formulation used. Even with identical Q and R weighting terms, the optimal trajectories will differ if a linear or bilinear model is used.

temperature set-point is not fairly contrasted with the cubic cost of the power required to generate a given amount of flow.

$$\begin{gathered}\min_{q(t)} \int_0^{t_f} [(T(t) - T_{set})^2 Q + Ru^2(t)]dt \\ \text{s.t.}\, q(t) \in \mathcal{q}, \forall\, t \in \left[0, t_f\right]\end{gathered} \tag{4.23}$$

The minimum-power thermal-limit protection objective (4.24) maintains the temperature within the safe temperature set over the fixed time horizon in such a way that minimizes the amount of power consumed by the controller $\mathcal{P}(t)$. Unlike the cost on u in the quadratic cost function, the functional cost on power is irrespective of the model formulation used to realize the controller. Instead, $\mathcal{P}(t)$ has a physical meaning as the number of Watts consumed by the pump in order to produce the flow rate $q(t)$ and depends on the amount of head (i.e. energy) loss through the pipe network to produce such a flow. See Section 3.2.3 for details on the empirical 3rd-order polynomial function mapping flow to power. If the system starts with a temperature outside $\mathcal{T}_{set}$, then the objective is undefined and another controller must be used to bring the temperature within the operating range of this controller.

$$\begin{gathered}\min_{q(t)} \int_0^{t_f} \mathcal{P}(x, u; t)dt \\ \text{s.t.}\left(T(t) \in \mathcal{T}_{set}; q(t) \in \mathcal{q}, \forall \left[0, t_f\right]\right)\end{gathered} \tag{4.24}$$

Similar to the minimum-power thermal-limit protection objective, the efficient one-sided regulation objective replaces the strict temperature constraint $T(t) \in \mathcal{T}_{set}$ with a soft ηth-order ($\eta \in \mathbb{Z}^+$) penalty on temperatures above T_{set}. This soft-constraint will allow the temperature to rise above the set point, but only in-so-much-as it offsets the additional ρ-weighted cost incurred by the additional power consumed.

$$\begin{gathered}\min_{q(t)} \int_0^{t_f} \left[\frac{1}{2}\left((T(t) - T_{set}) + |T(t) - T_{set}|\right) + \rho\, \mathcal{P}(x, u; t)\right] dt \\ \text{s.t.}\left(q(t) \in \mathcal{q}, \forall \left[0, t_f\right]\right)\end{gathered} \tag{4.25}$$

The following four subsections will derive the control policies that minimizes each of these objective functions for both linear and bilinear system models. In section 4.5 these control policies are calculated and applied to the simplified system under different configurations.

4.3.2.1 Minimum time

Given a time invariant system with state $x(t)$, control $u(t)$, and initial condition $x(0) = x_0$, the dynamics can be described by the general first-order vector nonlinear differential equation[1] $\dot{x}(t) = f(x, u; t)$. Assuming that the system begins outside the desired set $\mathcal{X}_f$, the goal of the minimum time controller is to drive the final state of the system $x(t_f)$ into $\mathcal{X}_f$ as quickly as possible given the constraints on the control $u(t) \in \mathcal{U}(x, u; t)\ \forall\ t \in \left[0, t_f\right]$. Since this chapter is only considering scalar systems, the problem is reduced to driving the state to the boundary of $\mathcal{X}_f$ if the state starts outside, otherwise no control is strictly necessary. Assuming that $\mathcal{X}_f$ includes the origin, the problem is further simplified to driving the state to $x_f \in \mathcal{X}_f$, which is the largest value within the set $\mathcal{X}_f$. This problem can be described by the following minimization problem (4.26) with the objective function defined by (4.27).

$$\min_{\substack{\dot{x}(t)=f(x,u;t) \\ u(t)\in\mathcal{U}(x,u;t) \\ x(t_0)=x_0 \\ x(t_f)\in\mathcal{X}_\mathrm{f} \\ 0<t_f}} \bar{J}(x, u; t) \tag{4.26}$$

$$\bar{J} = \int_0^{t_f} dt \tag{4.27}$$

The minimization's state constraint can be removed by augmenting the cost function $\bar{J}$ with a Lagrangian $p(t)$, yielding the new mini-max form (4.28) of the optimization and the new cost function (4.29).

[1] The notation $\dot{x}(t) = f(x, u; t)$ is an abbreviated notation for $\dot{x}(t) = f\big(x(t), u(t)\big)$. It describes a function f of two time-varying inputs $x(t)$ and $u(t)$, but the function f is time invariant, i.e. the variable t does not explicitly show up in f.

$$\min_{\substack{x(t) \\ u(t)\in\mathcal{U}(x,u;t) \\ x(0)=x_0 \\ x(t_f)=\mathcal{X}_f \\ t_f\geq 0}} \quad \max_{p(t)} J(x,u,p;t) \tag{4.28}$$

$$J = \int_{t_0}^{t_f} [\mathcal{H}(x,u,p;t) - p^T(t)\dot{x}(t)]dt \tag{4.29}$$

$$\mathcal{H}(x,u,p;t) = 1 + p^T(t)f(x,u;t) \tag{4.30}$$

Taking this new form, Appendix C derives Pontryagin's Minimum Principle specifying the necessary conditions for the optimal OL control trajectory $\boldsymbol{u}^* = \{u(t)\ \forall\ t \in [0,t_f]\}$ based on the Hamiltonian (4.30). These necessary conditions are:

1) $\mathcal{H}(x^*,u^*,p^*;t) \leq \mathcal{H}(x^*,u,p^*;t)\ \forall\ t \in [0,t_f] and\ \forall\ u \in \mathcal{U}(x,u;t)$

2) $\mathcal{H}(x^*,u^*,p^*;t) = 0\ \forall\ t \in [0,t_f]$

3) $\mathcal{H}_x = -\dot{p}^T = p^T f_x$

That is, 1) the Hamiltonian of the optimal control trajectory $\boldsymbol{u}^*$, the optimal state trajectory $\boldsymbol{x}^*$, and the optimal co-state trajectory $\boldsymbol{p}^*$ must be less than or equal to the Hamiltonian at each instant for $\boldsymbol{x}^*$, $\boldsymbol{p}^*$, and all other admissible control trajectories $\boldsymbol{u}$; 2) The Hamiltonian of the optimal trajectories $\boldsymbol{x}^*$, $\boldsymbol{p}^*$, and $\boldsymbol{u}^*$ must be equal to zero at each instant; and 3) the co-state must evolve as so, although no initial or final co-state need be specified.

Regardless of whether the control system analysis begins with a linear or bilinear formulation, the same optimal control law is derived in Appendix C. The optimal control system must operate in one of two linear modes, zero-flow or maximum-flow, resulting in a bang-bang type control. The resulting dynamics can be described by the hybrid automaton in Figure 4.3, with system matrices defined by (4.31), (4.32), and (4.33), in which the system switches from mode-**A** to **B** when the state touches the guards, $x \in \mathcal{G}(\boldsymbol{A},\boldsymbol{B})$. The optimal

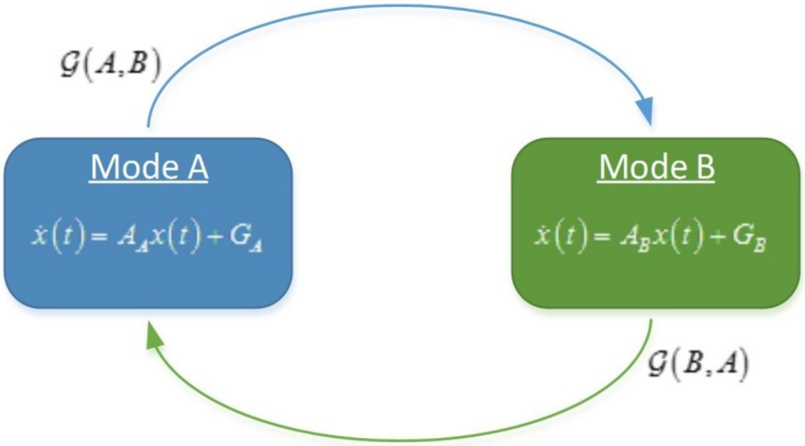


Figure 4.3 Hybrid system representation of controlled system

control problem then becomes defining the guards or switching surfaces, which force the switch from zero-flow to maximum-flow, or vise-versa.

Studying the dynamics of mode **A** results in (4.34) and reveals that the state will monotonically approach $x_{e_0} = \frac{-G_A}{A_A}$. Likewise, the dynamics of mode B in (4.35) reveal that the state monotonically approaches $x_{e_{max}} = \frac{-G_B}{A_B}$.

$$A_A = \frac{-h_a}{mc_p}; \quad G_A = \left(\frac{-1}{mc_p}\right)\left(h_a T_a - h_a T_{set} + \dot{Q}_h\right) - T_{set} \tag{4.31}$$

$$A_B = \frac{-h_{max} - h_a}{mc_p} \tag{4.32}$$

$$G_B = \left(\frac{-1}{mc_p}\right)\left((T_w - T_{set})h_{max} + (T_a - T_{set})h_a + \dot{Q}_h\right) - T_{set} \tag{4.33}$$

$$x(t) = \left(x_0 - x_{e_0}\right)e^{A_A t} + x_{e_0} \tag{4.34}$$

$$x(t) = \left(x_0 - x_{e_{max}}\right)e^{A_B t} + x_{e_{max}} \tag{4.35}$$

The direction of flow in the system with its dependence on operating mode and the values of x_{e_0} and $x_{e_{max}}$ is shown in Figure 4.4 below. Since $\{h_a, h_{max}, m, c_p\} \subset \mathbb{R}^+$, the system matrices have the property that $0 > A_A > A_B$ and $|G_A| > |G_B|$, which mean that $x_{e_0} > x_{e_{max}} > (T_w - T_{set})$ or $(T_w - T_{set}) > x_{e_{max}} > x_{e_0}$.

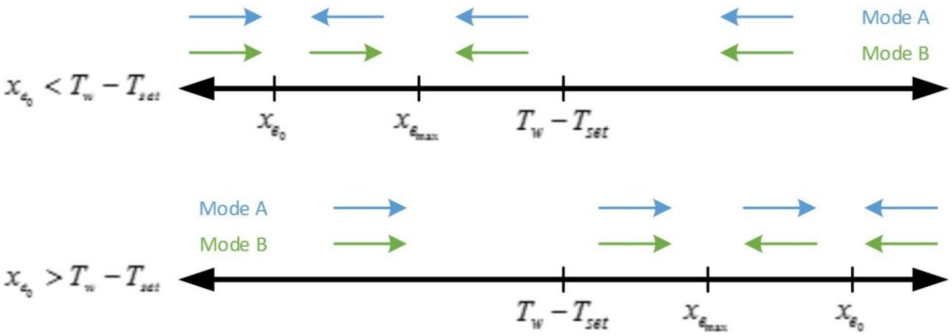


Figure 4.4 Flow diagram of mode A and B

Because the controlled system is only capable of monotonically approaching x_{e_0} or $x_{e_{max}}$ not all final states x_f are reachable from initial states x_0. Appendix C analyzes these scenarios and derives the switching surfaces $\mathcal{G}(\cdot,\cdot)$. Due to the simplicity of this system, a closed form equation of the state trajectory (4.36) can be obtained as a piece-wise continuous function that switches at-most one time (at t_1) from $t = 0 \dots t_f$ based on the relationship between x_0, x_{e_0}, $x_{e_{max}}$, and $\tilde{x} = T_w - T_{set}$.

$$x(t) = \begin{cases} (x_0 - x_{e_0})e^{(t)A_A} + x_{e_0}; & (0 \le t \le t_f) \wedge \begin{cases} \tilde{x} < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < \tilde{x} \end{cases} \\ (x_0 - x_{e_{max}})e^{(t)A_B} + x_{e_{max}}; & (0 \le t \le t_f) \wedge \begin{cases} x_{e_{max}} < \tilde{x} < x_f < x_0 \\ \tilde{x} < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < \tilde{x} \\ x_0 < x_f < \tilde{x} < x_{e_{max}} \end{cases} \\ (x_0 - x_{e_{max}})e^{(t)A_B} + x_{e\,max}; & (0 \le t \le t_1) \wedge \begin{cases} x_{e_0} < x_f < \tilde{x} < x_0 \\ x_0 < \tilde{x} < x_f < x_{e_0} \end{cases} \\ (\tilde{x} - x_{e_0})e^{(t-t_1)A_A} + x_{e\,0}; & (t_1 < t \le t_f) \wedge \begin{cases} x_{e_0} < x_f < \tilde{x} < x_0 \\ x_0 < \tilde{x} < x_f < x_{e_0} \end{cases} \\ {}^{\mathrm{N}}/_{\mathrm{A}}; & \mathrm{o.w.} \end{cases} \tag{4.36}$$

Where

$$t_1 = \frac{1}{A_B} \ln\left(\frac{\tilde{x} - x_{e_{max}}}{x_0 - x_{e_{max}}}\right) \tag{4.37}$$

$$t_f = \begin{cases} \frac{1}{A}\ln\left(\frac{x_f - x_{e_0}}{x_0 - x_{e_0}}\right) + t_0\,; & \begin{cases} \tilde{x} < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < \tilde{x} \end{cases} \\ \frac{1}{A_B}\ln\left(\frac{x_f - x_{e_{max}}}{x_0 - x_{e_{max}}}\right) + t_0; & \begin{cases} x_{e_{max}} < \tilde{x} < x_f < x_0 \\ \tilde{x} < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < \tilde{x} \\ x_0 < x_f < \tilde{x} < x_{e_{max}} \end{cases} \\ \frac{1}{A_A}\ln\left(\frac{x_f - x_{e_0}}{\tilde{x} - x_{e_0}}\right) + t_1; & \begin{cases} x_{e_0} < x_f < \tilde{x} < x_0 \\ x_0 < \tilde{x} < x_f < x_{e_0} \end{cases} \\ {}^{\mathrm{N}}/_{\mathrm{A}}\,; & \mathrm{o.w.} \end{cases} \tag{4.38}$$

This OL control solution specifies a valid control schedule when the initial condition satisfies any of the inequalities in the first three parts of (4.36). For all other initial conditions and system configurations, the control schedule is undefined because it is not possible to reach the desired safe set. In practice the control schedule described above would not be strictly followed. If pumping water raises the block temperature closer to the set-point, then no water should be pumped, thus saving energy. Additionally, this control algorithm is not guaranteed to be optimal, it has only been shown to meet the necessary, but not sufficient, Pontryagin's minimum conditions to be a locally optimal control. Once the final time has been reached, this control algorithm ceases to be appropriate and some other control algorithm must take over. For example, the flow could be set to a level that results in the controlled equilibrium state equal to the desired final state. This post-objective control, described by (4.39) and derived from (4.8), is only appropriate for a certain set of system parameters and final states where it is possible to maintain the state in equilibrium at the edge of the safe set.

$$\begin{gathered} \frac{\left(q(t)\right)^{\alpha_1}\alpha_0 T_w + h_a T_a + \dot{Q}_h}{h_a + \left(q(t)\right)^{\alpha_1}\alpha_0} = T_{set} \\ \therefore q(t) = \left(\frac{(T_{set} - T_a)h_a - \dot{Q}_h}{(T_w - T_{set})\alpha_0}\right)^{\frac{1}{\alpha_1}} \end{gathered} \tag{4.39}$$

Section 4.5 will use this OL controller for a variety of test scenarios in both simulation and experimentally. Additionally, the controller will be incorporated as the basis for an explicit MPC.

4.3.2.2 Quadratic regulation

Given the system (4.40) with scalar state x, scalar control u, and initial condition x_0, an admissible control trajectory $u(t) \in \mathcal{U}(x, u, t)$ from $t = 0 \dots t_f$ is to be applied that minimizes the sum, $\bar{J}$, of the integral quadratic regulation cost and the integral quadratic cost on control with weight $\rho \geq 0$. This can be described by the minimization problem (4.42).

$$\dot{x}(t) = f(x, u; t); \quad x(0) = x_0 \tag{4.40}$$

$$\bar{J} = \int_0^{t_f} [x^2(t) + u^2(t)\rho] dt \tag{4.41}$$

$$\min_{\substack{\dot{x}(t)=f(x,u;t) \\ u(t)\in\mathcal{U}(x,u;t) \\ x(0)=x_0}} \bar{J}(x, u; t) \tag{4.42}$$

The optimal programing problem (4.42) is in the class of problems that can be solved with the aid of the *calculus of variations* (Bryson and Ho 1975). The problem can be simplified by removing the constraint $\dot{x}(t) = f(x, u, t)$ and augmenting the cost function with the amount of constraint violation multiplied by the Lagrange multiplier $p(t)$. This yields the new augmented cost function J described by (4.43) where the instantaneous cost $L(x, u, t)$ is described by (4.44).

$$J = \int_0^{t_f} \left[L(x, u) + p(t)\big(f(x, u) - \dot{x}(t)\big)\right] dt \tag{4.43}$$

$$L(x, u) = x^2(t) + u^2(t)\rho \tag{4.44}$$

Integration by parts of (4.43) to get rid of the $\dot{x}(t)$ term yields (4.45) in which the Hamiltonian $\mathcal{H}$ is defined as (4.46). For simplicity's sake, the variation w.r.t. time will no longer be shown and will be implied from here forward for x, u, and p.

$$J(\boldsymbol{x},\boldsymbol{u},\boldsymbol{p}) = [-p\,x]_{t=t_f} + [p\,x]_{t=t_0} + \int_{t_0}^{t_f} [\mathcal{H}(x,u,p;t) + \dot{p}(t)(t)x(t)]dt \tag{4.45}$$

$$\mathcal{H}(x,u,p;t) = (x^2(t) + u^2(t)\rho) + p(t)f(x,u;t) \tag{4.46}$$

The minimization problem (4.42) can now be transformed into the min-max problem (4.47) with optimal trajectories $\boldsymbol{x}^*$, $\boldsymbol{u}^*$, $\boldsymbol{p}^*$ for the state, control and co-state respectively.

$$\min_{\substack{u(t)\in\mathcal{U}(x,u,t)\\ x(t)=x_0}} \max_{p(t)\geq 0} J(\boldsymbol{x},\boldsymbol{u},\boldsymbol{p}) \tag{4.47}$$

Using the calculus of variations to derive Pontryagin's Minimum Principle results in four necessary conditions for the optimal control trajectory (Kirk 2004):

1) $\dot{x}^* = f(x^*,u^*)\forall t \in [0,t_f], x^*(0) = x_0$

2) $\dot{p}^* = -\frac{\partial\mathcal{H}}{\partial x}(x^*,u^*,p^*)\forall t \in [0,t_f], p^*(t_f) = 0$

3) $\mathcal{H}(x^*,u^*,p^*) \leq \mathcal{H}(x^*,u,p^*)\forall u \in \mathcal{U}(x,u), t \in [0,t_f]$

4) $\mathcal{H}(x^*,u^*,p^*) = 0 \;\forall\; t \in [0,t_f]$

These four conditions can be used to derive a minimizing control (although not necessarily globally minimizing) in a way that is more computationally tractable than (4.42) because it consists only of two first-order differential equations and an optimization of u^* that is no longer coupled with values of u^* at previous and future times.

Adapting (4.46) and the four necessary conditions to the linear plant described by (4.9)…(4.15) yields the four necessary conditions for optimality described below with the reduced Hamiltonian defined as (4.48):

1) $\dot{x}_L{}^* = A_L x_L^* + B_L u_L^* + G_L \, \forall \, t \in [0, t_f]; \quad x_L(0) = x_{L_0}$

2) $\dot{p_L}^* = -A_L p_L^* - 2Q_L x_L^* \, \forall \, t \in [0, t_f]; \quad p_L(t_f) = 0$

3) $\min_{u_L \in g_L(x_L^*, u_L)} \bar{\mathcal{H}}_L(p_L^*, u_L) \ \forall \, t \in [0, t_f]$

4) $((x_L^*)^2 + (u_L^*)^2 \rho) + (A_L x_L^* + B_L u_L^* + G_L) p_L^* = 0 \, \forall \, t \in [0, t_f]$

$$\bar{\mathcal{H}}_L(p_L^*, u_L) = R_L u_L^2 + B_L p_L^* u_L \tag{4.48}$$

The Hamiltonian was reduced by removing terms in which alterations of the control would have no effect on the minimization. Per the derivation in Appendix C, if the system were unconstrained, (4.48) would be minimized by (4.49). However, since the system is constrained and $\bar{\mathcal{H}}_L$ is quadratic w.r.t. u, the minimizing u is $u_{L_{uc}}^*$ if it is admissible, otherwise one of the instantaneous bounds, 0 or $u_{L_{max}}$, is minimizing as described by (4.51).

$$u_{L_{uc}}^*(p_L^*) = \frac{-B_L p_L^*}{2R_L} \tag{4.49}$$

$$u_{L_{max}}(x_L^*) = -(x_L^* - T_w + T_{set}) h_{w_{max}} \tag{4.50}$$

$$u_L^* = \begin{cases} u_{L_{uc}}^*(p_L^*), & \text{if } C(x_L^*, u_L^*) \leq 0 \\ u_{L_{max}}^*(x_L^*), & \text{else if } \overline{\mathcal{H}_L}(p_L^*, u_{L_{max}}^*) < 0 \\ 0, & \text{o.w.} \end{cases} \tag{4.51}$$

Together the three conditions form (4.52), a two-point boundary value problem (TPBVP), the solution to which meets the necessary condition of the optimal trajectories X_L^*, U_L^*, and P_L^*. The optimal control that solves the TPBVP is the feedback law (4.53) that is a function of both the state and co-state. Because the dynamics of the state are only known forwards in time and the dynamics of the co-state are only known backwards in time, a special algorithm, discussed below, must be applied to solve the TPBVP and generate a minimizing OL control policy.

$$\begin{aligned} \dot{x}_L &= A_L x_L + B_L u_L + G_L; \quad x_L(0) = x_{L_0} \\ \dot{p}_L &= -A_L p_L - 2Q_L x_L; \quad\;\; p_L(t_f) = 0 \end{aligned} \tag{4.52}$$

$$u_L = \begin{cases} u_{L_{uc}}, & \text{if } C(x_L, u_L) \leq 0 \\ u_{L_{max}}, & \text{else if } \overline{\mathcal{H}_L}\big(p_L, u_{L_{max}}\big) < 0 \\ 0, & \text{o.w.} \end{cases}$$

$$\bar{\mathcal{H}}_L\big(p_L, u_{L_{max}}\big) = R_L u_{L_{max}}^2 + B_L p_L^* u_L \tag{4.53}$$

$$u_{L_{uc}}(p_L) = \frac{-(B_L p_L)^2}{4R_L}$$

$$u_{L_{max}}(x_L) = -(x_L - T_w + T_{set})h_{w_{max}}$$

Similarly, the necessary conditions for the optimal control of the bilinear plant described by (4.16)…(4.21) are listed below with the reduced Hamiltonian defined by (4.54):

1) $\dot{x}_{BL}^* = A_{BL}x_{BL}^* + (B_{BL}x_{BL}^* + b_{BL})u_{BL}^* + G_{BL} \; \forall \, t \in \big[0, t_f\big]; \; x_{BL(0)} = x_{BL_0}$

2) $\dot{p}_{BL}^* = -A_{BL}p_{BL} - B_{BL}u_{BL}p_{BL} - 2Q_{BL}x_{BL} \; \forall \, t \in \big[0, t_f\big]; \; p_{BL}\big(t_f\big) = 0$

3) $\underset{0 \leq u_{BL}^* \leq u_{BL_{max}}}{min} \; \bar{\mathcal{H}}_{BL}(x_{BL}^*, p_{BL}^*, u_{BL}) \; \forall \, t \in \big[0, t_f\big]$

4) $((x_{BL}^*)^2 + (u_{BL}^*)^2\rho) + (A_{BL}x_{BL}^* + (B_{BL}x_{BL}^* + b_{BL})u_{BL}^* + G_{BL})p_{BL}^* = 0 \; \forall \, t \in \big[0, t_f\big]$

$$\bar{\mathcal{H}}_{BL}(x_{BL}^*, p_{BL}^*, u_{BL}) = R_{BL}u_{BL}^2 + (B_{BL}x_{BL}^* + b_{BL})p_{BL}^* u_{BL} \tag{4.54}$$

The Hamiltonian was reduced by removing terms in which alterations of the control would have no effect on the minimization. If the system were unconstrained (4.54) would be minimized by (4.55). However, since the system is constrained and $\bar{\mathcal{H}}_{BL}$ is quadratic w.r.t. u, the minimizing u is $u_{BL_{uc}}^*$ if it is admissible, otherwise one of the bounds, 0 or $u_{BL_{max}}$, is minimizing as described by (4.56).

$$u_{BL}{}_{uc}^*(x_{BL}^*, p_{BL}^*) = \frac{-\big((B_{BL}x_{BL}^* + b_{BL})p_{BL}^*\big)^2}{4R_{BL}} \tag{4.55}$$

$$u_{BL}^{*} = \begin{cases} u_{BL}{}_{uc}^{*}(x_{BL}^{*}, p_{BL}^{*}), & \text{if } 0 \leq u_{BL}{}_{uc}^{*}(x_{BL}^{*}, p_{BL}^{*}) \leq u_{BL_{max}} \\ u_{BL_{max}}, & \text{else if } \bar{\mathcal{H}}\left(x_{BL}^{*}, p_{BL}^{*}, u_{BL_{max}}\right) < 0 \\ 0, & \text{o. w.} \end{cases} \tag{4.56}$$

Together the three conditions form (4.57), a two-point boundary value problem (TPBVP), the solution to which meets the necessary condition of the optimal trajectories X_{BL}^{*}, U_{BL}^{*}, and P_{BL}^{*}. The optimal control that solves the TPBVP is the feedback law (4.58) that is a function of both the state and co-state. Because the dynamics of the state are only known forwards in time and the dynamics of the co-state are only known backwards in time, a special algorithm, discussed below, must be applied to solve the TPBVP and generate a minimizing OL control policy.

$$\begin{gathered} \dot{x}_{BL} = A_{BL}x_{BL} + (B_{BL}x_{BL} + b_{BL})u_{BL} + G_{BL} \\ \dot{p}_{BL} = -A_{BL}p_{BL} - B_{BL}u_{BL}p_{BL} - 2Q_{BL}x_{BL} \\ x_{BL}(0) = x_{BL_0}; \quad p_{BL}\left(t_f\right) = 0 \end{gathered} \tag{4.57}$$

$$\begin{gathered} u_L = \begin{cases} u_{L_{uc}}, & \text{if } C(x_L, u_L) \leq 0 \\ u_{L_{max}}, & \text{else if } \bar{\mathcal{H}}\left(p_L, u_{L_{max}}\right) < 0 \\ 0, & \text{o. w.} \end{cases} \\ u_{BL_{uc}}(x_{BL}, p_{BL}) = \frac{-(B_L p_L)^2}{4R_L} \\ u_{L_{max}}(x_L) = -(x_L - T_w + T_{set})h_{w_{max}} \end{gathered} \tag{4.58}$$

The TPBVP that results from the linear and bilinear formulations can be solved using the methods presented in Section 4.3.3. After incorporating these two controllers into a MPC, Section 4.5 shows simulations and experimental results of system performance for a variety of test scenarios.

4.3.2.3 Minimum-power thermal-limit protection

The minimum-time controller puts no explicit cost on control effort (although control effort is in effect reduced by minimizing the amount of time any control is used) and the quadratic-cost controller places a different cost on control depending on whether the system is analyzed using linear or a bilinear formulation. In an effort to circumvent these issues, a minimum-power controller with thermal-limit protection is derived.

Given a generic first-order scalar system (i.e. in the form of (4.40)) with bounded control $u(t) \in \mathcal{U}(x, u; t)$, the minimum-power state-protection controller that solves the program (4.24). This solution should maintain the state x within the safe set $\mathcal{X}_s$ over a fixed prediction horizon t_f while minimizing control energy as defined by $\mathcal{P}(x, u; t)$ which is a monotonically increasing function w.r.t. the amount of flow $q(t)$ produced. $\mathcal{P}$ was empirically identified as a 3rd-order polynomial that passes through the origin in the form of $\mathcal{P}(q; t) = \sum_{i=1}^{3} a_i q^i(t)$ with non-negative coefficients a_i. The objective can be written as the optimal program (4.59) in which the objective cost $\bar{J}(\boldsymbol{x}, \boldsymbol{u})$ is defined by (4.60) as a function of the state trajectory $\boldsymbol{x} \coloneqq \{x(t): t \in [0, t_f]\}$ and control trajectory $\boldsymbol{p} \coloneqq \{p(t): t \in [0, t_f]\}$.

$$\min_{\substack{\dot{x}(t)=f(x,u;t) \\ u(t)\in\mathcal{U}(x,u;t) \\ x(t)\in\mathcal{X}_s\ \forall\ t\in[0,t_f] \\ x(0)=x_0}} \bar{J}(\boldsymbol{x}, \boldsymbol{u}) \tag{4.59}$$

$$\bar{J}(\boldsymbol{x}, \boldsymbol{u}) = \int_0^{t_f} \mathcal{P}(x, u; t)\, dt \tag{4.60}$$

Similar to the derivation of the quadratic controller, the program (4.59) can be simplified by removing the state evolution constraint and augmenting the cost function with the amount of constraint violation multiplied by the Lagrange multiplier $p(t)$. Additionally, a second Lagrange multiplier $\lambda(t)$ must be added that ensures the state constraint is satisfied. The state constraint is captured by the new ancillary state $\nu(t)$ that obeys (4.61) which is zero whenever the state-constraint is satisfied, and positive whenever violated. This yields the new min-max program (4.64) as a function of the augmented cost (4.63) simplified through integration by parts, and the Hamiltonian described by (4.65). Multiplying the new state $\nu(t)$ by the co-state λ means that the constraint will be satisfied in the solution of the program (4.64); otherwise as ν tries to minimize J the amount of constraint violation can be infinitely magnified by λ.

$$\begin{aligned}\dot{\nu}(t) &= C^2(x;t)\mathbb{1}\big(-C(x;t)\big) \\ &= x^2(t)\mathbb{1}\big(-x(t)\big) \\ &= \max\big(0, x^2(t)\big)\end{aligned} \tag{4.61}$$

$$\mathbb{1}(x) = \begin{cases} 0, & x \leq 0 \\ 1, & x > 0 \end{cases} \tag{4.62}$$

$$\begin{aligned} J(\boldsymbol{x},\boldsymbol{p},\boldsymbol{\lambda},\boldsymbol{u}) = [-p^T x]_{t=t_f} + [p^T x]_{t=t_0} \\ + \int_0^{t_f} [\mathcal{H}(x,u,p,\lambda;t) + \dot{p}^T x] dt \end{aligned} \tag{4.63}$$

$$\min_{\substack{u(t)\in\mathcal{U}(x,u;t) \\ x(t)\in\mathcal{X}_S\ \forall\ t\in[0,t_f] \\ x(0)=x_0}} \max_{\substack{p(t) \\ \lambda}} J\,(\boldsymbol{x},\boldsymbol{p},\boldsymbol{\lambda},\boldsymbol{u}) \tag{4.64}$$

$$\mathcal{H}(x,u,p,\lambda;t) = \mathcal{P}(x,u;t) + p(t)f(x,u;t) + \Big(x^2(t)\mathbb{1}\big(-x(t)\big)\Big)\,\lambda(t) \tag{4.65}$$

In a fashion similar to previous sections, a new augmented cost function (4.63), a min-max problem (4.64), and a necessary conditions for an optimality can be derived. The necessary conditions for the optimal control, state, and co-states trajectories, $\boldsymbol{u}^*$, $\boldsymbol{x}^*$, $\boldsymbol{p}^*$, and $\boldsymbol{\lambda}^*$ using Pontryagin's minimum principle, are as follows:

1) $\dot{x}^*(t) = f(x^*, u^*; t)\ \ \forall\ t \in [0, t_f]; \quad x^*(0) = x_0$

2) $\dot{p}^*(t) = -\mathcal{H}_x(x^*, u^*, p^*, \lambda^*; t)\ \forall\ t \in [0, t_f]; \quad p^*(t_f) = 0$

3) $\dot{\lambda}^* = 0 \Rightarrow \lambda^*(t) = \lambda^* = \text{constant}\ \forall\ t \in [0, t_f]$

4) $\mathcal{H}(x^*, u^*, p^*, \lambda^*; t) \leq \mathcal{H}(x^*, u, p^*, \lambda^*; t)\ \forall\ u \in \mathcal{U}(x, u; t), t \in [0, t_f]\ \forall\ t \in [0, t_f]$

5) $\mathcal{H}(x^*, u^*, p^*, \lambda^*; t) = 0\ \forall\ t \in [0, t_f]$

Unlike the previous nonlinear programs for the controllers derived in Section 4.3.2.1 and Section 4.3.2.2, the five necessary conditions for (4.64) do not form a TPBVP. The states x are defined at the beginning and the co-states p at the end, but the value of λ while constant, is not defined anywhere. Thus, the TPBVP solvers presented in Section 4.3.3 cannot be used for (4.64). Instead, three other methods will be briefly described here, but due to their computational intractability or poor assurance on convergence, they will not be applied to solve problems in this study. The first method could be to remove λ and ν from the program and instead use a gradient descent method (briefly described in Section 4.3.3) to optimize the u trajectory. This method requires a modification of the control at each instance that the state is on the boundary, which instantaneously drives the state away from the boundary. The problem being, the modified control to drive the state away from the boundary, may not always be feasible. Similarly using the gradient descent method, a second method could be used that optimizes over the u trajectory and the value of λ, but the value of λ is only updated iteratively after each time u has converged. This method could have significant computational complexity due to iterating back-and-fourth between λ and u. Additionally, λ could only ever increase, so if an incorrect $\boldsymbol{u}$ forced a jump in λ, the algorithm could never recover to a lower optimal value of λ. Lastly, a dual method could be used, which would optimize over the co-state trajectory $\boldsymbol{p}$ instead of the control trajectory, but first, it must be proven that there is an equivalency to the solution to (4.64) and its dual. Due to the complications with all three of these algorithms, the Minimum-power thermal-limit protection controller will not be implemented. Instead, the Efficient Cooling Regulator will be derived next.

4.3.2.4 Efficient Cooling Regulator

Replacing the hard state constraints in the previous controller with a soft constraint removes the need for the ancillary state. If the constraint is quantified by (4.68) the ‘firmness’ of the constraint can be increased by increasing the value of $\eta \in \{2,4,6 \dots\}$. This penalty is added to the power consumed by the controller $\mathcal{P}(x,u;t)$ in the cost function $\bar{J}(\boldsymbol{x},\boldsymbol{u})$ defined by (4.67) as part of the minimization problem defined by (4.66).

$$\min_{\substack{\dot{x}(t)=f(x,u;t)\\ u(t)\in\mathcal{U}(x,u;t)\\ x(t_0)=x_0}} \bar{J}(\boldsymbol{x},\boldsymbol{u};t) \tag{4.66}$$

$$\bar{J} = \int_{t_0}^{t_f} \left[\mathcal{R}^{\eta}\big(x(t)\big) + \rho\,\mathcal{P}(x,u;t)\right]dt \tag{4.67}$$

$$\begin{gathered}\mathcal{R}^{\eta}(x) = \left(\frac{1}{2}(x+|x|)\right)^{\eta} = \big(x\,\mathbb{1}(x)\big)^{\eta}\\ \text{where } \mathbb{1}(x) = \begin{cases}0, & x \le 0\\ 1, & x > 0\end{cases}\end{gathered} \tag{4.68}$$

A constraint can be removed from the nonlinear program (4.66) to simplify it by augmenting the cost function $\bar{J}(\boldsymbol{x},\boldsymbol{u})$ with a Lagrangian $p(t)$ yielding the new augmented cost function $J(\boldsymbol{x},\boldsymbol{u},\boldsymbol{p})$ and the new min-max program (4.69).

$$\min_{\substack{u(t)\in\mathcal{U}(x,u;t)\\ x(t)\\ x(0)=x_0}} \max_{p(t)} J(\boldsymbol{x},\boldsymbol{u},\boldsymbol{p}) \tag{4.69}$$

$$J(\boldsymbol{x},\boldsymbol{u},\boldsymbol{p}) = \int_{0}^{t_f} [\mathcal{H}(x,u,p;t) - p(t)\dot{x}(t)]dt \tag{4.70}$$

$$\mathcal{H}(x,u,p;t) = \mathcal{R}^{\eta}\big(x(t)\big) + \rho\,\mathcal{P}(x,u;t) + p(t)f(x,u;t) \tag{4.71}$$

Proceeding as in previous sections, there are four necessary conditions for the optimal control trajectory $\boldsymbol{u}^*$ that solves (4.68), along with the optimal state and co-state trajectories $\boldsymbol{x}^*$ and $\boldsymbol{p}^*$.

1) $\dot{x}^*(t) = f(x^*,u^*;t)\ \forall\, t \in [0,t_f]; \quad x^*(0) = x_0$

2) $\dot{p}^*(t) = -\mathcal{H}_x(x^*,u^*,p^*,\lambda^*;t)\ \forall\, t \in [0,t_f]; \quad p^*(t_f) = 0$

3) $\mathcal{H}(x^*,u^*,p^*;t) \le \mathcal{H}(x^*,u,p^*;t)\ \forall\, u \in \mathcal{U}(x,u;t), t \in [0,t_f]$

4) $\mathcal{H}(x^*,u^*,p^*,\lambda^*;t) = 0\ \forall\, t \in [0,t_f]$

Before adapting (4.71) and the four conditions above for use with the linear plant described by (4.9)…(4.15), it is necessary to define the power usage w.r.t. u. Equation (4.10) was used to transform the empirical 3rd order monotonic function to a form w.r.t. x_L and u_L yielding (4.72) in which the parameters $\{a_1, a_2, a_3\} > 0$. This function passes through the $\mathcal{P}_L$-x_L-u_L origin, is positive for all admissible u_L, is monotonically increasing w.r.t. u_L when $x_L < (T_w - T_{set})$, and is monotonically decreasing for $x_L > (T_w - T_{set})$.

$$\mathcal{P}_L(x_L, u_L) = \begin{cases} 0, & x_L = T_w - T_{set} \\ \sum_{i=1}^{3} \left[\left(\frac{-u_L}{(x_L - T_w + T_{set})\alpha_0} \right)^{\frac{i}{\alpha_1}} a_i \right], & \text{o.w.} \end{cases} \quad (4.72)$$

The four necessary conditions for optimality of the control, state, and co-state trajectories for the linear system are listed below as a function of the Hamiltonian (4.73), t the variation of the ramp function w.r.t. x_L (4.74), the control power consumed (4.75), and the variation of the power w.r.t. x_L (4.76).

$$\mathcal{H}_L(x_L, u_L, p_L) = \mathcal{R}_L^{\eta}(x_L) + \rho\, \mathcal{P}_L(x_L, u_L) + f_L(x_L, u_L)\, p \quad (4.73)$$

$$\left(\mathcal{R}_L^{\eta}(x_L) \right)_x = \eta\, x_L^{\eta - 1} \mathbb{1}(x_L) \quad (4.74)$$

$$\mathcal{P}_L(x_L, u_L) = \begin{cases} 0, & \begin{array}{l} x_L = T_w - T_{set} \\ \text{or } u_L = 0 \end{array} \\ \sum_{i=1}^{3} \left[\left(\frac{-u_L}{(x_L - T_w + T_{set})\alpha_0} \right)^{\frac{i}{\alpha_1}} a_i \right], & o.w. \end{cases} \quad (4.75)$$

$$\mathcal{P}_{L_x}(x_L, u_L) \\ = \begin{cases} 0, & \begin{array}{l} x_L = T_w - T_{set} \\ \text{or } u_L = 0 \end{array} \\ \sum_{i=1}^{3} \left[\left(\frac{a_i\, \alpha_0\, i}{\alpha_1\, u_L} \right) \left(\frac{-u_L}{(x_L - T_w + T_{set})\alpha_0} \right)^{\frac{\alpha_1 + i}{\alpha_1}} \right], & \text{o.w.} \end{cases} \quad (4.76)$$

1) $\dot{x}_L^* = A_L x_L^* + B_L u_L^* + G_L\ \forall\, t \in [0, t_f]; \quad x_L^*(0) = x_0$

2) $\dot{p}_L^* = -\left(\mathcal{R}_L^{\eta}(x_L^*) \right)_x - \rho\, \mathcal{P}_{L_x}(x_L^*, u_L^*) - A_L p\ \forall\, t \in [0, t_f]; \quad p_L^*(t_f) = 0$

3) $u_L^* = \underset{u_L \in \mathcal{U}_L(x_L, u_L)}{\operatorname{argmin}}\ \mathcal{H}_L(x_L^*, u_L, p_L^*)\ \forall\, t \in [0, t_f]$

4) $\mathcal{H}_L(x_L^*, u_L^*, p_L^*) = 0 \; \forall \; t \in [0, t_f]$

A derivation of the feedback control law would show that the control is zero, ($u_L^* = 0$) when (4.77) is satisfied. Otherwise the optimal control is the solution to the convex program (4.78) that can be solved numerically using efficient computer programs (e.g. gradient descent or simplex algorithms).

$$(x_L^* - T_w + T_{set})\, p_L^* \leq 0 \tag{4.77}$$

$$u_L^* = \underset{u_L \in \mathcal{U}_L(x_L, u_L)}{\operatorname{argmin}} \left[\rho\, \mathcal{P}_L(x_L^*, u_L) + B_L u_L p_L^*\right] \tag{4.78}$$

With the ability to compute a numerical solution to the feedback law (4.78), and the first and second necessary conditions above, a TPBVP can be formulated similar to the one found in Section 4.3.2.2. Numerical methods for solving these types of TPBVPs will be presented in Section 4.3.3.

The necessary optimality conditions for bilinear systems evolving according to (4.16)…(4.21) are listed below, and are functions of the bilinear Hamiltonian (4.79), the variation of the ramp function w.r.t. x_{BL} (4.80), and the empirically-derived monotonically-increasing control power consumption (4.81).

$$\mathcal{H}_{BL}(x_{BL}, u_{BL}, p_{BL}) = \mathcal{R}_{BL}^{\eta}(x_{BL}) + \rho\, \mathcal{P}_{BL}(u_{BL}) + f_{BL}(x_{BL}, u_{BL})\, p \tag{4.79}$$

$$\left(\mathcal{R}_{BL}^{\eta}(x_{BL})\right)_x = \eta\, x_{BL}^{\eta-1} \mathbb{1}(x_{BL}) \tag{4.80}$$

$$\mathcal{P}_{BL}(u_{BL}) = \sum_{i=1}^{3} \left[\left(\frac{u_{BL}}{\alpha_0}\right)^{\frac{i}{\alpha_1}} a_i\right] \tag{4.81}$$

1) $\dot{x}_{BL}^* = A_{BL} x_{BL}^* + (B_{BL} x_{BL}^* + b_{BL}) u_{BL}^* + G_{BL} \; \forall \; t \in [0, t_f]; \quad x_{BL}^*(0) = x_0$

2) $\dot{p}_{BL}^* = -\left(\mathcal{R}_{BL}^{\eta}(x_{BL}^*)\right)_x - (A_{BL} + B_{BL} u_{BL}^*)\, p \; \forall \; t \in [0, t_f]; \quad p_{BL}^*(t_f) = 0$

3) $u_{BL}^* = \underset{u_{BL} \in [0, h_{w,max}]}{\operatorname{argmin}} \mathcal{H}_{BL}(x_{BL}^*, u_{BL}, p_{BL}^*) \; \forall \; t \in [0, t_f]$

4) $\mathcal{H}_{BL}(x_{BL}^*, u_{BL}^*, p_{BL}^*) = 0 \; \forall \; t \in [0, t_f]$

Due to the non-quadratic form of $\mathcal{P}_{BL}$ it is not possible to analytically derive the optimal feedback law w.r.t. x_{BL}^* and p_{BL}^*, except when $(b_{BL} > 0)$ and $(x_{BL}^* < 0)$ in which case $u_{BL}^* = 0$. Instead of analytical methods, the solution to the third condition above can be numerically calculated as the solution to the simplified convex minimization problem (4.82).

$$u_{BL}^* = \underset{u_{BL} \in [0, h_{w,max}]}{\operatorname{argmin}} \left[\rho \, \mathcal{P}_{BL}(u_{BL}) + (B_{BL} x_{BL} + b_{BL}) u_{BL}\right] \tag{4.82}$$

Similar to the linear case shown above, a control trajectory that satisfies the conditions for optimality can be calculated as the solution to the TPBVP of the state dynamics forward in time, the co-state dynamics backwards in time, and a feedback control function that is a solution to a nonlinear, but convex, minimization problem. Section 4.3.3 will present algorithms for solving for such trajectories.

The computational complexity of realizing the linear and bilinear efficient cooling regulators shown in this section will be greater than the complexity of the TPBVPs presented in Section 4.3.2.2 because of the iterative minimization methods required to solve the feedback algorithm. The advantage being that both the linear and bilinear cases shown in this section should produce same optimal control trajectory since the cost function is irrespective of the dynamic model form used. This consistency of solutions should give a more fair comparison of the computational complexity of using linear versus bilinear dynamic models, thus allowing for the most efficient model and solver to be implement in the embedded wireless control system.

4.3.3 Solving the TPBVP

There are three main methods to solving the TPBVPs: gradient projections, shooting methods, and first-order gradient methods (Kirk 2004). The gradient projection method does not use any of the necessary conditions from Pontryagin's Minimum Principle and instead optimizes the entire control trajectory accounting for the effect each u has on the control at future instances. Shooting methods solve the TPBVPs by starting with a guess for the initial value of the co-state, integrate the state and co-state forward in time using the feedback control law, then iteratively adjust the guess for the co-state initial value in order to satisfy the constraint $p(t_f) = 0$. First-order gradient algorithms solve the TPBVPs by starting with a guess of the control trajectory at discrete instances in time that do not necessarily obey the feedback law. Then, that control trajectory is used to integrate the state equation forward in time to get the state trajectory, followed by integrating the co-state equation backwards in time to get the co-state trajectory. With both the state and co-state at each instance in time, the feedback law is used to identify which direction the control trajectory should be perturbed at each instance in time to more closely satisfy the necessary condition on the next iteration.

It would be difficult to synthesize an on-line embedded controller using the gradient projection method due to the large memory requirements associated with optimizing over the time-coupled control trajectory. While the shooting method has minimal memory requirements, an ideal feature for embedded applications, it is not particularly numerically stable for control problems like the ones solved here. The issue arises when the control can bang between the limits causing minor changes in the co-state initial value guess to result in large fluctuations in the final value of the co-state. Fortunately, the first-order gradient method balances numerical stability with a memory requirement of $O(MK + NK)$ where M, N, and K are the number of control inputs, the number of states, and the number of steps chosen to discretize the control horizon, respectively.

The four main steps executed during each iteration i of the first-order gradient algorithm are (Kirk 2004)

1) Select the number of steps K to discretize the control horizon and generate a guess of control values $u^{(0)}(t_k)$ at each step $k = 0 \dots K - 1$. An arbitrary guess can be used for the initial iteration, but when wrapped in a MPC the control trajectory at the previous time of control action will serve as a good estimate of the current trajectory.

2) Use the current estimate of the optimal control trajectory $U^{(i)}$ to integrate the state equations from $t = 0 \dots t_f$ with initial condition $x(0) = x_0$, storing the value of this updated state trajectory estimate $\boldsymbol{x}^{(i)}$.

3) Use the current estimate of the control and state trajectory $\boldsymbol{u}^{(i)}$ and $\boldsymbol{x}^{(i)}$ respectively to integrate the co-state equation backwards in time from the final state $p\left(t_f\right) = 0$. Using the values of $x^{(i)}(t_k)$ and $p^{(i)}(t_k)$ to compute the value of $\hat{u}^{(i)}(t_k)$ that satisfies the feedback law. Store this trajectory of admissible controls.

4) If the difference between each $u^{(i)}(t_k)$ and $\hat{u}^{(i)}(t_k)$ is less than some predetermined ϵ, then terminate the iterative procedure yielding the minimizing control trajectory $\hat{\boldsymbol{u}}^{(i)}$. Otherwise, perturb each value of $u^{(i)}(t_k)$ towards the value of $\hat{u}^{(i)}(t_k)$ by some percentage τ, then iterate the whole procedure until the accuracy desired by ϵ is achieved. If τ is too large then oscillations can occur in the optimization, but if τ is too small then convergence may not occur in an adequate timeframe. Therefore, Kirk proposes that τ be adjusted after each iteration: slightly increased (e.g. 1%) if the cost J descreased from the previous iteration, or more significantly decreased (e.g. 5%) if the cost J increased from the previous iteration.

Assuming that an optimal solution exists, and that only a single solution satisfies the necessary conditions for optimality, and the above algorithm converges to a solution that satisfies the conditions, then the solution found is the optimal OL control solution. Solving for an OL control trajectory in this way can be implemented by identifying the system matrices and current state, running the TPBVP solver, then change the control at discrete instances in time to the appropriate value of $u(t_k)$. One problem that could arise is if the system matrices slowly vary (e.g. if the amount of heat injections or the water temperature changes). The initially calculated OL would no longer be optimal. To overcome this issue, the OL trajectory could be regularly updated as the system changes; this is the fundamental concept behind model predictive control (MPC) discussed next.

4.4 Model-Predictive control (MPC) of BLS

Feedback control, a.k.a. closed-loop (CL) control, theory is a diverse interdisciplinary field combining mathematics and engineering to manipulate the dynamic response of a system. The six main feedback control strategies; adaptive control, hierarchical control, intelligent control, optimal control, robust control, and stochastic control, are often combined to cause the response of the system to efficiently track a desired reference. The general block diagram for all these strategies is shown in Figure 4.5. When the desired reference is time varying the control design is termed a tracking problem, while a time invariant reference is termed a regulation problem. The primary control strategy considered in the proposed research is optimal control where a controller optimizes a cost index, which is a function of the dynamic response of the system and the amount of control effort. In order to specify a controller that optimizes a function of the response of the plant, a model of the plant must be known which can be used to predict the response.

In an optimal model predictive controller (MPC), also called a receding horizon controller, an open-loop optimal control sequence is calculated at regular periods in time. Between update intervals, the OL control trajectory is applied to the system. The OL control is optimal if it minimizes the predicted value of the objective function over the prediction horizon. Figure 4.6 shows the procedure executed by the MPC each time the OL trajectory $\boldsymbol{u}$ is updated

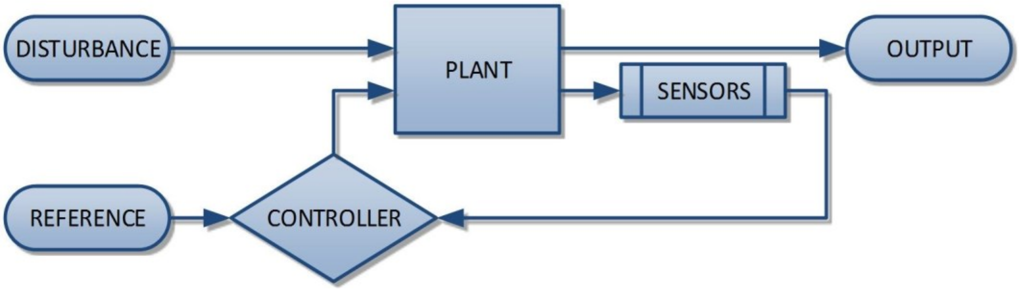


Figure 4.5 Feedback control block diagram

for a system were control input is limited and the system is trying to reach a reference trajectory. An iterative optimization process starts with a constant control $\boldsymbol{u}^{(0)}$ from the previous step, iterates and drives the predicted output closer to the reference with control $\boldsymbol{u}^{(2)}$, then finally achieves the optimal control $\boldsymbol{u}^{(2)}$ and state $\boldsymbol{x}^{(\mathbf{2})}$ trajectories on the second iteration. The following control instant a new optimal sequence is computed to reflect changes in the sensors, systems, or reference.

The other main optimal control strategy, linear quadratic regulation (LQR), works in a similar manner, but doesn't require online computation of the optimal control sequence; thus reducing computation demand on the controller. However, with MPC, unlike LQR, constraints on plant states and/or controller outputs are explicitly definable. This ability to handle constraints is especially important to nonlinear and constrained control problems often found in large-scale civil control problems. Model predictive control has seen great success in the process control industries since the 1980's with the advent of digital microcontrollers. Due to its long history, relevant literature

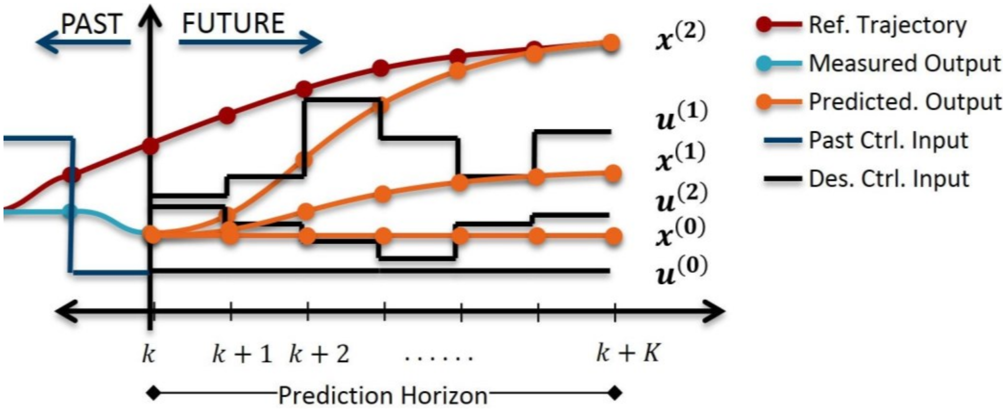


Figure 4.6 Graphical representation of MPC step

has extensively covered key topics such as stability, optimality, and robustness for linear and nonlinear plants (Bemporad et al. 1999; Camacho and Bordons 1999; Mayne et al. 2000; Morari and Lee 1999).

4.4.1 Model-predictive control fundamentals

The fundamental concept of MPC is the serial generation of OL control trajectories that minimize the predicted cost of an objective function. The MPC general formulation used through this dissertation assumes the dynamics of the system to be controlled can be described by the state $x(t) \in \mathbb{R}^n$ of the first-order vector differential equation (4.83) where $u(t) \in \mathbb{R}^m$ is the control input and $w(t) \in \mathbb{R}^p$ is the external disturbance at each instance in time. Since the MPC will be executed on a digital computer implementing a discrete time numerical integration routine, it makes sense to approximate (4.83) with the difference equation (4.84) which exhibits the Markov property and models the value of the state $x(k+1)$ at the next step $(k+1)$, equivalent to time $(t+\Delta t)$, given the control input $u(k)$ and external disturbance $w(k)$ at step k, equivalent to time t.

$$\dot{x}(t) = f_c(x, u, w; t) \tag{4.83}$$

$$x(k+1) = f_d(x, u, w; k) \tag{4.84}$$

The controller must possess a method for predicting the state trajectory of the system (4.85) from the current step k to the end of the prediction horizon K steps long, given the future control input trajectory (4.86), the estimated external disturbance trajectory (4.87), and the current value of the state $x(k)$. The goal of the controller is to minimize the integral cost (4.88) over the prediction horizon w.r.t. the control trajectory $\hat{\boldsymbol{u}}$ such that the control at each step is admissible, i.e. $u(k) \in \mathcal{U}(x, u; t)$.

$$\hat{\boldsymbol{x}} = \{\hat{x}(k+1) \dots \hat{x}(k+K)\} \tag{4.85}$$

$$\hat{\boldsymbol{u}} = \{\hat{u}(k+1) \dots \hat{u}(k+K)\} \tag{4.86}$$

$$\hat{\boldsymbol{w}} = \{\hat{w}(k+1) \dots \hat{w}(k+K)\} \tag{4.87}$$

$$\boldsymbol{J}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}, \hat{\boldsymbol{w}}) = \sum_{l=k+1}^{k+K} J\big(x(l), u(l), w(l)\big) \tag{4.88}$$

If the system dynamics (4.84) are linear, e.g. (4.9), and there are no constraints, i.e. $\mathcal{U} \coloneqq \mathbb{R}^m$, then a closed form function can be analytically calculated *a priori* and used online. Otherwise, some sort of numerical optimization must execute at each time the OL is updated. The optimization algorithms could be as simple as a linear program (LP) or require a more complex quadratic program (QP), sequential quadratic program (SQP), nonlinear program (NLP), or mixed integer NLP (MINLP) algorithm.

The MPC of BLS has been studied theoretically and applied to a variety of fields, leading to a handful of predominant methods. The system and objective can be linearized around the set-point. This method has been applied to the control of active sludge processes by (Ekman 2005). A more advanced linearization was proposed by (Han 1977; Nocedal et al. 1999) which adds a convex quadratic term to the cost that approximates the Hessian of the Lagrangian function. Sequential quadratic programming can be used to solve this type of approximate system, as was done by (Kelman and Borrelli 2011) for control of HVAC systems. Feedback linearization can be used to generate a feedback control function that linearizes the system w.r.t. to a new control that is the input to the feedback linearization function (Del Re et al. 1993). This enables the use of efficient linear MPC algorithms for nonlinear systems, but constraints on the control must be carefully handled during the design process. Feedback linearization was applied by (Piñón et al. 2005) to the temperature control of greenhouses which can be modeled as a BLS. All of these methods rely on some sort of approximation of the system in order to reduce the computational complexity of the optimization algorithm that must run in real-time, at the cost of sub-optimal performance. Alternatively, it is possible to forego the linearization approximations and solve the TPBVP, that results from the necessary conditions for an optimal solution, directly using methods similar to (Graichen and Käpernick 2011). Leveraging the latest in dual-core floating point MCUs and the relative simplicity of the studied scalar hydronic control problem, this dissertation is interested in pursuing such a method.

4.5 Application of BLS MPC to the hydronic test bed

In section 4.3.2 four MPC objectives were described, and three were derived. These three controllers, in both their linear and bilinear forms (except for the minimum-time which has identical linear and bilinear control laws) have been coded and implemented on the *Martlet* wireless controller presented in Chapter 2 and installed on the test bed described in section 3.2. Because closed form performance metrics cannot be derived for all three MPC, three different empirical test cases were used to test the performance of the different MPC formulations in both numerical simulation and experimentally.

4.5.1 Embedding bilinear MPC on the *Martlet*

The first-order gradient procedure for solving TPBVPs outlined above was codified in the C language, integrated into the *Martlet* firmware, compiled using Code Composer Studio v5 ("Code Composer Studio" 2013) and embedded into the *Martlet* wireless controller. The GNU Scientific Library (GSL) (Galassi and Theiler 2013) was ported to run on the *Martlet* in its native single floating point precision. The GSL Runge-Kutta 4-5 ODE solver was used to forward integrate the state and backwards integrate the co-state. The GSL one dimensional minimization routines using the Brent minimization algorithm was used to solve the convex optimization required at each step along the prediction trajectory as part of computing the optimal control w.r.t. the state and co-state for the efficient cooling controller of Section 4.3.2.4. The TPBVP solver was written as an extension to GSL using the same architecture as a generic algorithm that is passed function pointers for the application specific functions. Care was taken when writing the MPC routines to run as fast as possible by minimizing the use of nested functions, passing of large variables (instead pointers to structures are used), and recasting of structures. Each MPC step is sped up via a 'warm start' technique, i.e. the previous control trajectory, shifted in time by one step, is used as an initial guess for the TPBVP solver. Two different MPC techniques were considered, one in which the online optimizer runs until convergence is reached, no matter how long that takes; and another modeled after the 'fast MPC' (Wang and Boyd 2010) where the optimization runs until either convergence is reached or until a certain amount of time has elapsed. Although the 'fast MPC' is not guaranteed to converge at every time step, and thus the resulting control may not be optimal, the faster update rate may perform better than the convergence guaranteed MPC.

Tuning of the controllers was done empirically through trial and error in simulation. Table 4.1 shows the non-model based parameters for the minimum time (MT) controller, the MPC with quadratic state and control costs with a linear model (QL), the MPC with quadratic state and control costs with a bilinear model (QBL), the efficient cooling MPC of section 4.3.2.4 with a linear model (LEC), and the efficient cooling MPC of section 4.3.2.4 with a bilinear model (BLEC). All the controllers have a $500s$ prediction horizon with 100 steps that are $5s$ long and assumed convergence was reached when the RMS change in the control trajectory was less than ε_u =5e-3 or $i_{max} = 250$ iterations were reached.. The value of ρ (the same as R if $Q = 1$ in the quadratic controllers) was selected to have the desired performance. The TPBVP solver's rate of descent τ is bounded by τ_{min} and τ_{max} and initialized as τ_0 . Values of τ were chosen such that convergence was reached in approximately 100 steps. The linear controllers were particularly sensitive to smaller τ_{min} values or larger τ_{max} values which prevented convergence either due to oscillation or slow descent. The GSL ODE solver and minimizer also have ε parameters for stopping criteria that were set to the largest values that produced results comparable to the results generated by the equivalent MATLAB functions with the default parameters.

Table 4.1
Controller Parameters Used for Testing

Controller	ρ	η	τ_{min}	τ_0	τ_{max}	ε_u	K	i_{max}
MT	-	-	-	-	-	5E-3	100	250
QL	3E-2	-	5E-3	2E-2	2E-1	5E-3	100	250
QBL	1	-	2E-2	5E-2	2E-1	5E-3	100	250
LEC	1	2	2E-3	1E-1	4E-1	5E-3	100	250
BLEC	1	2	2E-2	5E-2	2E-1	5E-3	100	250

Table 4.2
Test Case Parameters

	$T_0\ (^\circ C)$	$T_{set}\ (^\circ C)$	$T_w\ (^\circ C)$	$T_a\ (^\circ C)$	$\dot{Q}_h\ \left({}^J/_s\right)$
(1) Cooling	60.5	40	24	24.8	0
(2) Heating	21	30	20	22	35
(3) Switching	~36	40	~22	~24	switches

The next two sub-sections describe in more detail how these parameters were selected, and how the five different controllers compare against one-another on a variety of metrics.

4.5.2 Test scenarios

In order to empirically test the performance of the five controllers in Table 4.1, three different test cases were used in both simulation and experimentally. Summarizing from Table 4.2, Case (1) starts above the set-point temperature; Case (2) starts below but heat is applied that will raise the temperature; and Case (3) starts below then switches the heater on for 10 minutes, off for 5 minutes, on for 30s, off for 30s, on for 1 minute, off for 1 minute, on for two minutes, and then off.

Case (1) and (2) were primarily used for tuning the open-loop controller calculated at each MPC step, resulting in the values in Table 4.1. Once the OL controller performance was satisfactory, it was wrapped into a MPC that was able to handle the changes in system properties seen in Case (3) when the heater switches on and off.

4.5.3 Results

The results will be presented in the following way: First Case (1) and (2) are run in MATLAB in order to arrive at appropriate values for the OL controller parameters and determine which controllers should be studied further. Case (1) will be used to verify that those parameters on the selected controllers produce the same OL control trajectory when run on the *Martlet* using GSL. Finally Case (3) was experimentally run on the demonstrator using a further subset of the best performing MPC as both a fully convergent MPC and 'fast MPC'.

The state and control[1] trajectories at each iteration of the TPBVP solver for the QL, QBL, LEC, and BLEC OL controllers for test Case (1) are shown in Figure 4.7…Figure 4.10 respectively. The first iteration is shown in a dark blue color and the line color progresses through the color spectrum until the color red at the 250th iteration (ε_u was set to 0 during the convergence test). The QL, LEC, and BLEC controller reach the final value well before 250 iterations as seen by the red line, while the QBL controller takes more iterations in order achieve the sharp edges of the final trajectory. If however slight sub-optimality can be tolerated, then the QBL controller can be terminated earlier, resulting in sub-optimal control when switching from the maximum flow to zero flow. For all of the objective functions and models, the rate of convergences varied and was negatively affected by the complexity of the optimal control trajectory and the distance from the initial guess.

[1] Remember that the value of the control trajectory has a different physical meaning for the controllers with linear and bilinear models.

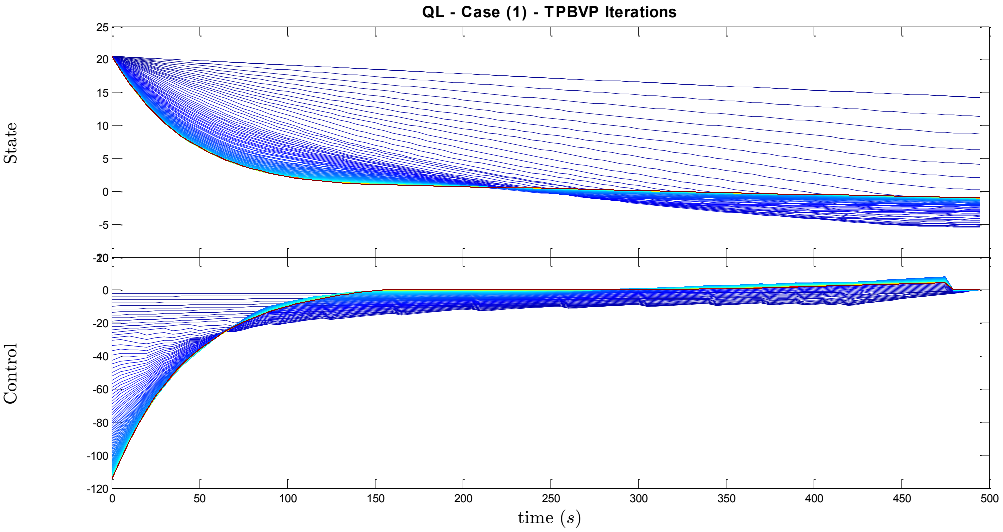


Figure 4.7 TPBVP convergence analysis for quadratic cost and linear model

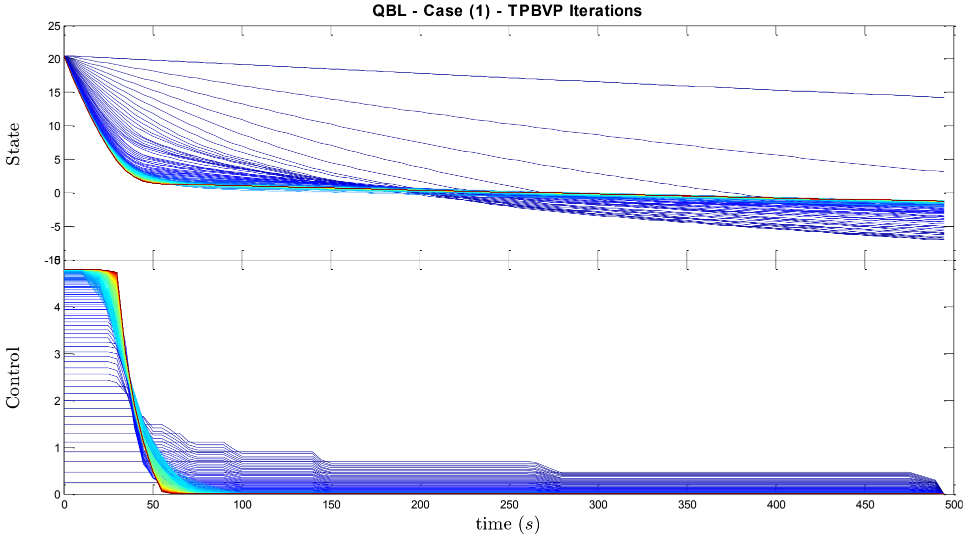


Figure 4.8 TPBVP convergence analysis for quadratic cost and bilinear model

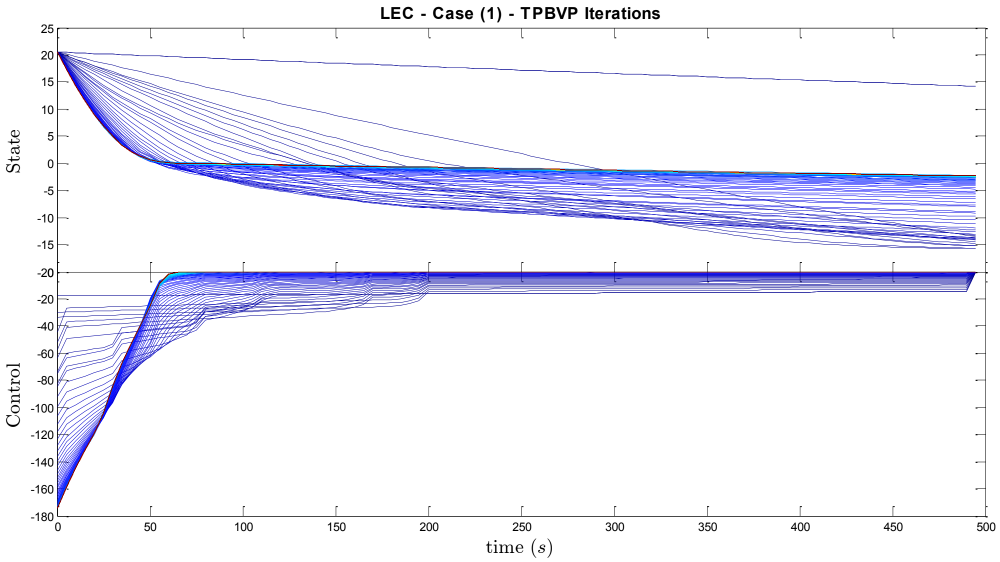


Figure 4.9 TPBVP convergence analysis for efficient cooling with linear model

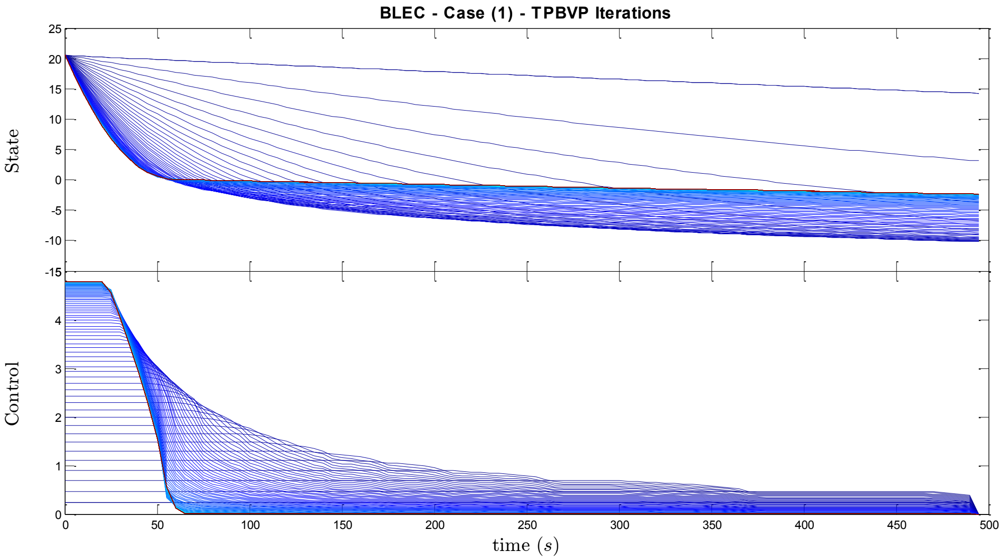


Figure 4.10 TPBVP convergence analysis for efficient cooling with bilinear model

Figure 4.11 shows the resulting block temperature and flow trajectories that result from the final trajectories in Figure 4.7…Figure 4.10. Additionally the zero flow (pink line), maximum flow (green line), and minimum time controller (blue line) are shown as reference benchmark controllers. As expected, the minimum time controller tracks the maximum flow until the set point is reached. The QL controller does not cool as well as the others partially because the control penalty does not have the desired physical meaning, and partially because smaller values of R require more than double the amount of iterations to solve the TPBVP. The QBL controller performs as desired, matching closely to the minimum-time controller. The lines for the LEC and BLEC controllers lie on top of each other, meaning that the two converge to the same solution, unlike the QL and QBL controllers. This is desirable because it allows for an even comparison between MPC formulations using linear versus bilinear models. The circles illustrate selected points along the trajectory computed by the *Martlet*. While the *Martlet's* values do not perfectly match MATLAB simulations (shown as solid lines) due to the lower numerical precision on the *Martlet*, the results are similar enough for practical purposes.

The MATLAB simulations for Case (2) resulted in the block temperature and flow trajectories shown in Figure 4.12. Similar to Case (1), the MT, QBL, LEC, and BLEC controllers produce similar state trajectories. The LEC and BLEC controllers once again produce the same control trajectory, within the tolerances of the TPBVP solver. The QL controller once again does not perform as desired due to similar reasons as above.

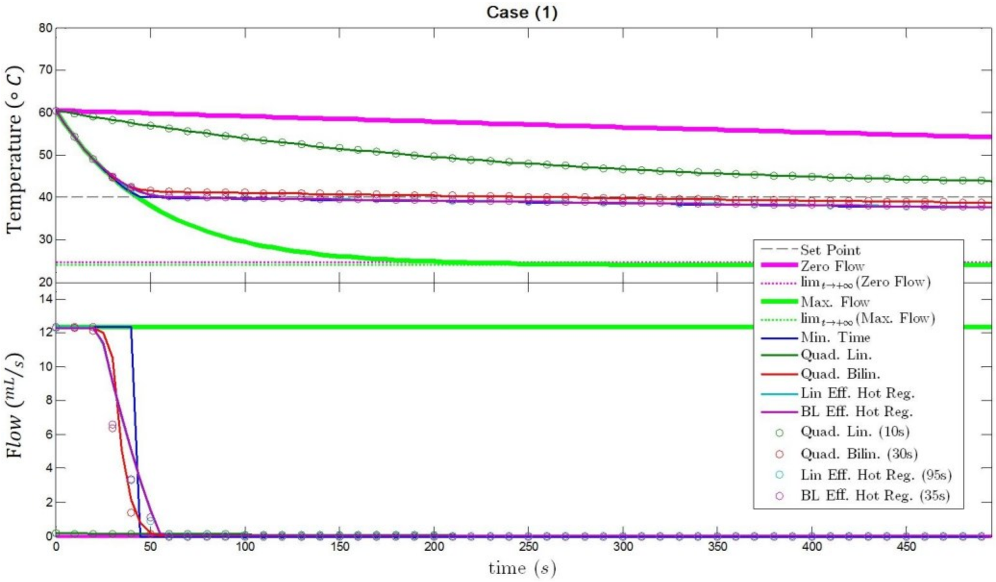


Figure 4.11 Case (1) MATLAB and *Martlet* OL control simulation

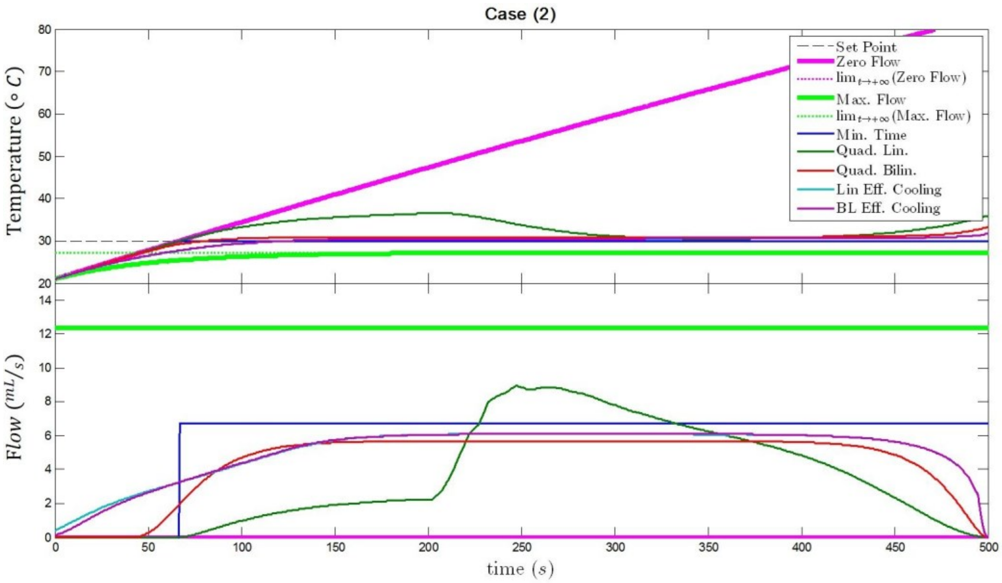


Figure 4.12 Case (2) MATLAB OL control simulations

While Table 4.3 compares the cost incurred by each controller for Case (1), an even comparison is not easily made. Each controller is trying to minimize the its own objective function and therefore cannot be expected to perform as well as another controller using a different objective. However the LEC and BLEC can be compared in this way since they both are using the objective, and result in the same trajectory. In this case, the other metric of interest is how computationally complex the controller is, when measured by runtime of the OL control optimization. For both the quadratic and efficient cooling controllers, the controllers with bilinear models converged more quickly.

Table 4.3
Summary of Results from OL Simulations for Case (1)

Controller	State Cost[1]	Control Cost[1]	Total Cost	Runtime[2]
QL	2120	1909	4029	10.3s
QBL	1274	169	1443	8.3s
LEC	1222	100	1322	29.4s
BLEC	1225	100	1325	14.8s

[1] These costs do not have an equivalent meaning across controllers, except for the LEC and BLEC

[2] The runtime was measured in MATLAB using the `tic` and `toc` commands

Out of the four controllers from the previous analyses, the BLEC controller was selected for further study since it ran faster than the LEC with the same result. The BLEC controller was also desirable because it directly penalizes the power consumed by the pump and only places a cost on block temperature when it is too high. The minimum time controller was selected as the benchmark due to its simplicity and good performance. The following three figures show the experimental results of the MT, BLEC MPC, and BLEC 'fast MPC' controllers running on a *Martlet* when implemented on the simple system with one pump and one thermal load. The green shadow in the figures indicated when the heater was turned on (producing $35W$ of heat) during the experiment, and the circles and x's indicate when the MPC first started computing (a circle indicates the TPBVP converged and an x means that it did not). The final trajectory resulting from each online optimization is then shown as a dotted line that starts dark blue and slowly fades to red through the spectrum from the first MPC step at $t = 0s$ until the final MPC step at the end of the thick black line of experimental measurements.

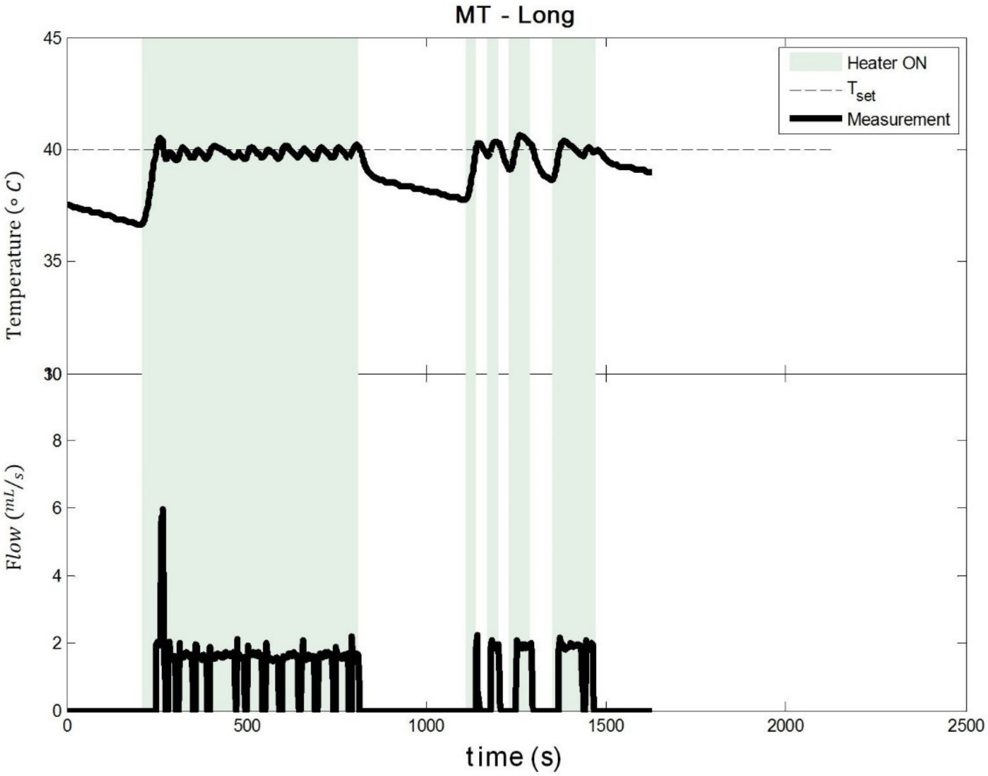


Figure 4.13 Case (3) minimum-time control experimental response

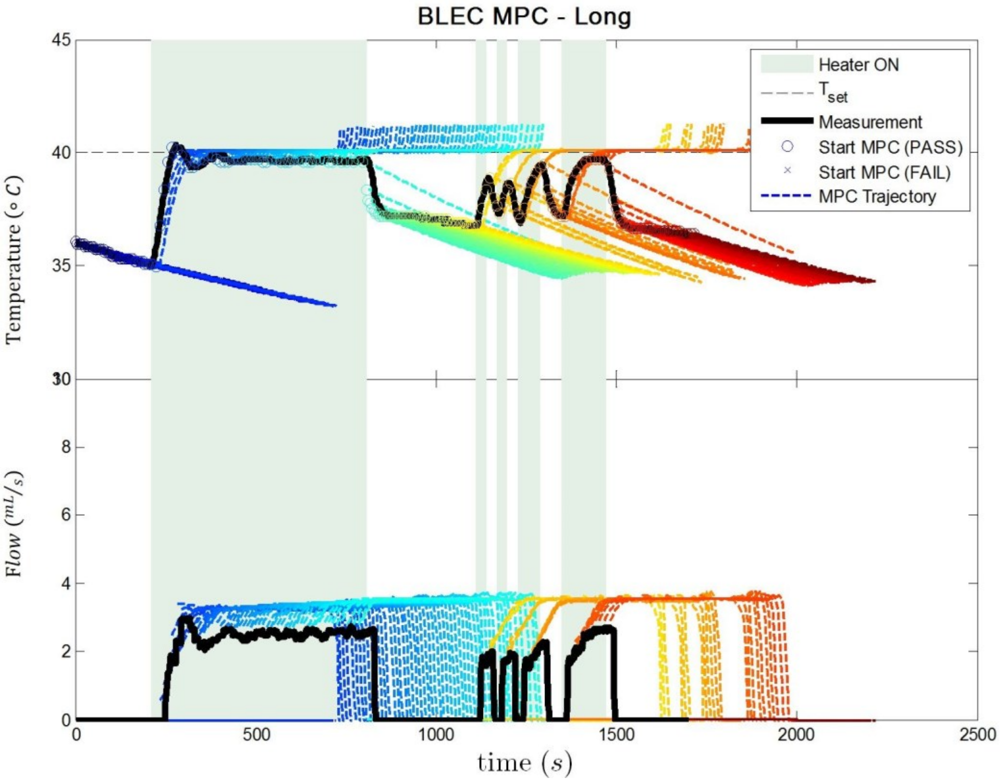


Figure 4.14 Case (3) BLEC convergent MPC experimental response

Figure 4.13 shows the experimental response of the minimum time controller which keeps the temperature pretty much just at the set-point, but exceeds it in a few spots just after the heater turns on, i.e. it has an undesirable overshoot since the controller does not account for future costs of its current decisions. The MPC in which the TPBVP runs until convergence was experimentally implemented on the demonstrator producing the trajectories in Figure 4.14. By inspecting the spacing of the circles it can be seen that the TPBVP takes longer to converge when the heater switches on or off. This delay sometimes results in overshoot of the set point because the controller cannot react fast enough. In Figure 4.15 there are many x's when the heater switches on or off, meaning the TPBVP doesn't converge, but reaches a solution close enough that the performance is more desirable then waiting until convergence. In both Figure 4.14 and Figure 4.15 the desired flow rates (the beginning of the dotted lines) and the measured flow rates (the thick black line) do not exactly match, and thus performance could be affected. However,

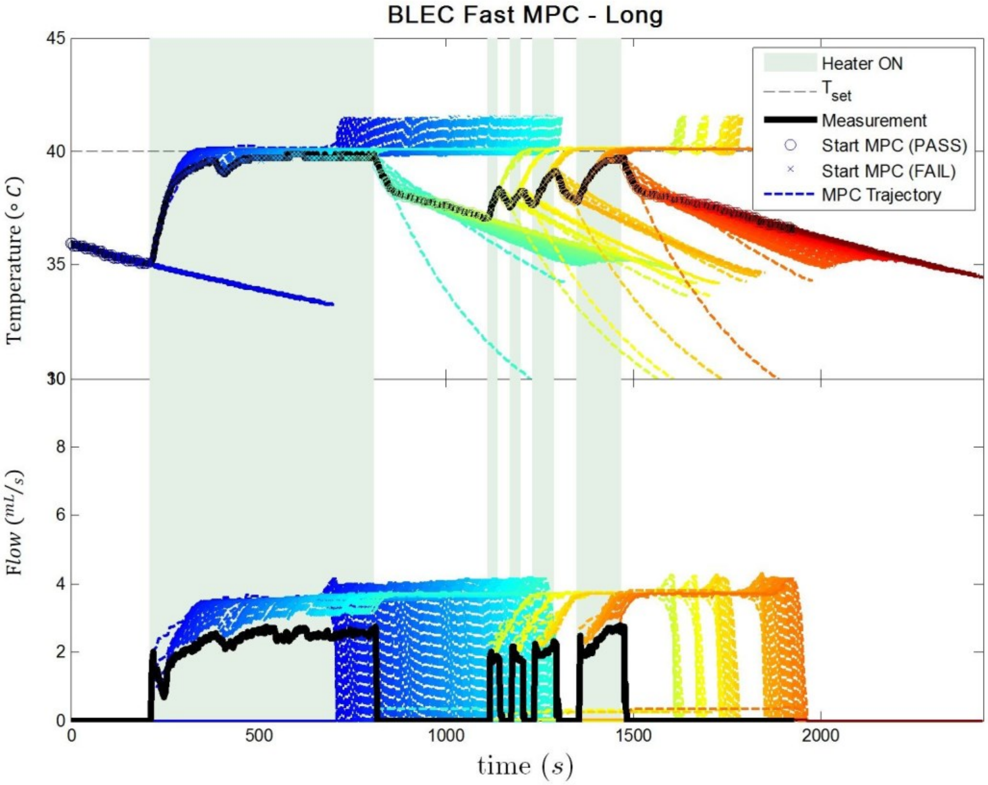


Figure 4.15 Case (3) BLEC 'fast MPC' experimental response

the MPC is able to stay on top of any divergences in expected system response and account for these model deficiencies when compared against an *a priori* computed OL control trajectory.

This chapter has focused on developing an effective MPC for control of a simple hydronic system. Linear and bilinear model forms were considered, along with three different control architectures with increasing computational complexity: minimum-time control, quadratic control, and efficient cooling control. Although more computationally costly than all but the LEC, the BLEC controller has performed exemplary in that it keeps the block temperatures below the threshold while consuming minimum pump power. Additionally it is capable of running using 'fast MPC' techniques on the *Martlet* quickly enough to follow the system dynamics. Therefore, the BLEC controller will be further considered as this dissertation continues. The later usages will extend these developments to a larger more complex hydronic system which will require BLEC of multiple blocks using multiple pumps and valve configurations.

4.6 EMBEDDED BILINEAR CONTROL SYSTEMS CONCLUSIONS

In this chapter bilinear systems were shown as a viable model for designing and implementing model predictive controllers for hydronic subsystems in cyber-physical infrastructure. Background and theory on BLS was presented, along with examples of the many real-life infrastructure systems which can be modeled with bilinear systems, including hydronic systems. A simple hydronics system, reduced from the test bed in Chapter 3, was presented for testing this chapter's developments. It was shown that for systems such as the simple hydronic system, there is more than one mathematically equivalent way to model the control system: a linear plant model with state-dependent control constraints and a bilinear plant model with time-invariant rectangular control constraints. These two methods, both common in literature, were analyzed for their applicability to control hydronic subsystems in cyber-physical infrastructure. A key metric was their empirical computational and memory complexity, important for execution on a low-power MCU such as the one on the *Martlet*. Similarly three objective functions were analyzed: a simple minimum-time objective, a quadratic cost, and an objective that explicitly met the design goal of efficient cooling.

This multi-pronged study lead to the determination that bilinear models with explicit efficient cooling objective functions would be best suited for embedding into a wireless node such a *Martlet*. Experimental analysis of the controller's performance further validated the *Martlet* as an effective platform and showed that the proposed controller could replace more simple suboptimal controllers often used in literature. These promising outcomes present an advancement in MPC controllers run on wireless nodes and could open up new areas in wireless CPS previously thought unreachable.

Chapter 5.

Hybrid Dynamics and Distributed Control of CPS

In this chapter the MPC algorithms introduced in Chapter 4, that were designed for simplified control systems with a single point of actuation (input) and a single controlled measurement (output) (SISO), are extended to a control complex cyber-physical infrastructure systems. These more complex systems have multiple control inputs and multiple control outputs (MIMO), discontinuities in the dynamics and controls, and networked communication between nodes. The paradigm of hybrid systems, which exhibit both continuous and discrete dynamics, is presented as an approach to designing controllers for many different types of cyber-physical infrastructure systems. Additionally, the wireless control paradigm introduced in Chapter 2 is posed as a technology for implementing such advanced infrastructure control systems. In the middle of this chapter, the challenges and opportunities of an agent-based control (ABC) approach utilizing wireless sensing and control are described. After covering the opportunities and challenges of the hybrid approach and a paradigm of wirelessly networked agent-based control systems, the hydronics test bed of Chapter 3 is reintroduced as a motivating example. Combining the MPC algorithms of Chapter 4 with hybrid control theory and wireless ABC theory yields a controller for the hydronics test bed that is implemented on a network *Martlets* and tested experimentally. The advantages and challenges of this approach are compared with other approaches at the columniation of this chapter.

5.1 Introduction to Hybrid Dynamical Systems (HDS)

The interaction of continuous with discrete dynamics in a system is the principle concern of the field of hybrid dynamical systems (HDS) theory. Figure 5.1 shows a hybrid dynamical system with scalar or vector valued continuous inputs $u(t)$, discrete inputs $v(t)$, continuous outputs $y(t)$, discrete outputs $w(t)$, continuous states $x(t)$, and discrete states $q(t)$. A mathematical model of hybrid systems, building on elements of continuous and discrete models, includes the following elements (Lunze and Lamnabhi-Lagarrigue 2009):

- $\mathcal{X}$ continuous state space (flow set), typically $\mathcal{X} \in \mathbb{R}^n$
- $\mathcal{Q}$ discrete state space (jump set), such as $\mathcal{Q} = \{0, 1, 2, \dots, Q\}$
- $\boldsymbol{f}$ vector fields mapping the continuous dynamics for each $q \in \mathcal{Q}$ (flow map)
- (q_0, x_0) initial values of hybrid state
- δ discrete state transition function (jump map)
- $\mathcal{G}$ set of guards triggering discrete state transitions (*i.e.* restrictions on the discrete dynamics)

and can optionally include the following:

- $\mathcal{R}$ continuous state reset map after discrete state transitions
- Inv mode invariants of $\boldsymbol{f}$ (*i.e.* restricts evolution of continuous variables)

For the problems studied under the proposed approach, it is sometimes easier to consider the extension of continuous systems with discrete dynamics, instead of *vice versa*. Consider first the possibility that the vector field $\boldsymbol{f}$ may change discontinuously and autonomously. For example, such a change is present in thermostat controlled heating of a room where the

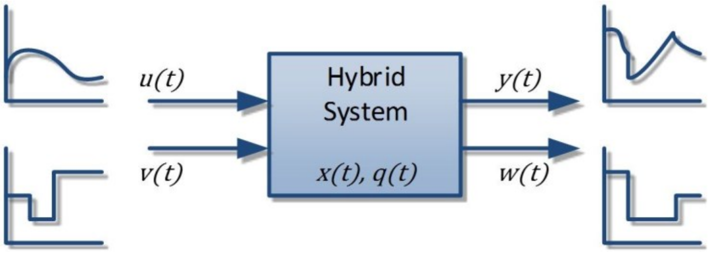


Figure 5.1 Input-output schematic of a hybrid dynamical system

temperature $x(t)$ dynamics are described by the switch at a threshold temperature T_{max} between two continuous dynamics described by (5.1).

$$\dot{x}(t) = \boldsymbol{f}\big(x(t)\big) \coloneqq \begin{cases} f_1\big(x(t)\big), & if\ x(t) \leq T_{max} \\ f_2\big(x(t)\big), & if\ x(t) > T_{max} \end{cases} \tag{5.1}$$

It is also possible for the switching to be *time-driven*, as opposed to *event-driven*, such as when system dynamics change according to a scheduled clock. Similarly, *time-driven* or *event-driven* autonomous state jumps can occur. One condition under which an autonomous state jump can occur is defined by the reset map $\mathcal{R}$. For example, the reset map, $\big(x(\bar{t}^-), x(\bar{t}^+)\big) \in \mathcal{R}$, causes the state to jump at time $\bar{t}$ from its position before the jump at $x(\bar{t}^-)$ to $x(\bar{t}^+)$ after the jump. Discrete dynamics and state jumps may also be controlled. This general template for hybrid dynamical systems lays the foundation for a wide variety models: hybrid automata, switched systems, piecewise affine models, timed automata, hybrid Petri nets, differential automata, mixed logical dynamical models, real-time temporal logics, complementarity systems, and hybrid inclusions, which are all extensively covered in (Lunze and Lamnabhi-Lagarrigue 2009).

5.1.1 Applications of the HDS approach

Physical systems with both continuous and discrete dynamics can arise in a multitude of ways. Additionally, HDS arise often when a continuous physical plant is controlled with a discrete input, updated at discrete intervals in time, or exhibits actuator saturation. Goebel, Sanfelice, and Teel, in their IEEE Control Systems Magazine article, describe some of the more common sources of hybrid phenomena such as colliding masses, impulsive behavior in biological systems, electrical circuit power factor calculation, sample-and-hold control systems, and hybrid controllers for continuous nonlinear systems (Goebel et al. 2009). The example HDS can be grouped into four categories, autonomous switching between continuous dynamics, autonomous jumps in a continuous state, controlled switching of continuous dynamics, or controlled continuous state jumps (Lunze and Lamnabhi-Lagarrigue 2009). A bouncing ball, e.g. within a pinball machine, switches between continuous dynamics before the impact with the surface, during the collision, and after the ricochet. An idealized collision of two hard objects, described by position and velocity states, exhibits a jump in the velocity state at the moment of the instantaneous collision. Controlled switching occurs in systems where the controller is either on or off, electrical switching in DC-DC converters or opening and closing of valves in fluid systems for example. Controlled state jumps occur in systems with continuous time clocks that are reset by the controller.

Within the realm of civil engineering, the different HDS models can be applied to a variety of different cyber-physical infrastructure systems. For example, a heater's thermostat can be extended to include all HVAC systems where the continuous state vector includes the temperature, humidity, CO_2, etc. in each zone, the discrete states are fan speeds, damper positions, and (re-)heating/cooling coil states. The continuous model for each zone, typically simplified as an R-C network, may have discrete dynamics caused by the opening and closing of doors and windows. Another example of a hybrid civil system, commonly used in the literature, is the dual-reservoir problem shown in Figure 5.2. A controller must discretely turn the valves on or off creating a prescribed flow through V 3.

Assuming only on-off operation of the valves in the dual-reservoir problem, there are $2^5 = 32$ discrete controllable states, plus 32 additional discrete states depending on the autonomous switching caused by the level of the water in the upper reservoir which allows water to flow over the open

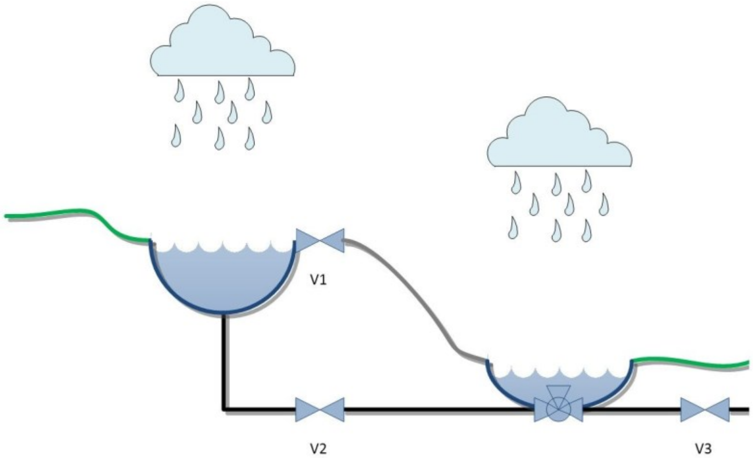


Figure 5.2 Hybrid system example: Dual Reservoir

sluice gate *V1*. Therefore the space of discrete states is $Q = \{1, 2, \dots, 64\}$. The dynamics of the two continuous states, $\mathcal{X} \coloneqq$ the height of the two reservoirs, depends on the discrete state of the system, meaning $\boldsymbol{f}$ maps $\boldsymbol{f}: Q \times \mathbb{R}^2 \to \mathbb{R}^2$. When the height of the upper reservoir reaches *V1* and *V1* is open (the condition that *V1* is open is a guard $\mathcal{G}$ on the discrete state transition), the limits of the invariant *Inv* for the given mode are reached and the discrete state jumps according to the discrete transition map δ. The reset mapping of the continuous variables after the change in discrete mode is unity and therefore negligible.

Other civil systems with hybrid dynamics include pipe networks where controlled switching occurs due to the valve, and uncontrolled switching may be observed due to leakage caused by pipe damage. The structural control problems exhibit continuous dynamics until yielding of structural members occur due to high displacements and some actuators exhibit discrete switching dynamics at saturation.

5.1.2 Modeling HDS

Due to the breadth of applications and phenomena that HDS theory covers, there are several different architectures that can be used to model hybrid behavior: hybrid automata, switched systems, piecewise affine systems, mixed logical-dynamical systems, complementarity systems, discretely controlled continuous systems, timed automata, and hybrid inclusions (Lunze and Lamnabhi-Lagarrigue 2009). **Hybrid automata** models extend the concept of the finite-state machines, often used in computer science, which model purely discrete behavior. Finite-state machines exist in only one of a finite number states at any given time, with a set of rules describing when and where to the state switches. The extension is made by introducing a continuous dynamical system within each state in which a continuity (or jump rule) must exists between the continuous states when switching from one discrete state to another. **Switched systems** are described by the differential equation (5.2) where $x(t)$ is the state vector and $f_{q(t)}(\cdot)$ defines the evolution of the continuous states when operating in discrete mode $q(\boldsymbol{x}, t)$. The model is said to be *piecewise smooth* if the switching occurs on a hyper-surface dependent only upon the current continuous state, illustrated by the two-discrete-state system (5.3). **Piecewise affine systems** are a subclass of piecewise smooth systems in which the continuous state evolution function and switching condition are affine w.r.t. the state and control. The affine restriction simplifies the mathematics of the analysis, leading to a larger body or research and more thorough proofs of controller stability and robustness.

$$\dot{x}(t) = f_{q(\boldsymbol{x},t)}\big(x(t)\big) \tag{5.2}$$

$$\dot{x}(t) = f\big(x(t)\big) = \begin{cases} f_-\big(x(t)\big), & \text{if } \phi\big(x(t)\big) < 0, \\ f_+\big(x(t)\big), & \text{if } \phi\big(x(t)\big) > 0. \end{cases} \tag{5.3}$$

Mixed logical dynamical systems possess state vectors with both continuous and Boolean (i.e. 1 or 0) components. An inequality is applied to the states, translating them into propositional logic describing the continuous and discrete evolution of the system. **Complementarity systems** are described by (5.4) and (5.5), the typical continuous state space system equations, and (5.6), a complementarity condition which models the discrete mode changes by forcing either $z_i(t) = 0$ or $w_i(t) = 0$ at any given instant.

$$\dot{x}(t) = f\big(x(t), w(t), u(t)\big) \tag{5.4}$$

$$z(t) = g\big(x(t), w(t), u(t)\big) \tag{5.5}$$

$$0 \le z(t) \perp w(t) \ge 0 \tag{5.6}$$

Similar to switched systems, **discretely controlled continuous systems** switch their operating mode when the continuous state crosses a hyper-surface in the state-space. Unlike switched systems, the mode of operation is not strictly analogous to the current subset of the state-space, e.g. a new mode can be engaged on the same side if the new mode forces it to 'bounce' off the switching-surface. **Timed automata** are a subset of hybrid automata in which the only continuous state obeys $\dot{x} = 1$, making them well suited for modeling systems with a clock. Lastly, **hybrid inclusions** extend the differential inclusion model $\dot{x} \in F(x)$ with added invariants, guards, and resets, which created a model well suited for modeling networked control systems.

Each of these models has its own niche of application areas where it finds use. Further complicating HDS theory, there is no 'cure all' approach that can be applied to model any system, and many of these models can be transformed into another model type (Lunze and Lamnabhi-Lagarrigue 2009). Choosing an appropriate model is a balance between broad applicability of the model (i.e. how far can the current system be abstracted yet still be modeled with such a model) and the ability to analyze the model and yield meaningful results with reasonable effort. In some cases depending on the desired accuracy, it is even possible through abstraction to model a HDS as either a purely continuous or discrete dynamical system, e.g. high-frequency DC-DC converters and air-flow through a diesel engine, respectively. Regardless of the type of model used, many problems can plague analytical analysis of HDS. Instability can occur in switched systems, even if stable behavior is expected from each of the subsystems between switches. In additional to stability and robustness concerns that continuous control designers are familiar with, HDS can exhibit Zeno behavior, i.e. when an infinite number of switching events occur in a finite-length time-interval. This behavior is famously seen in an ideal bouncing ball. Sliding modes, a similar problem to Zeno behavior, can occur in switched systems when the vector fields on both sides of the switching surface point towards the surface leading to infinitely fast switching that cannot be modeled with traditional continuous time integration algorithms. Lastly, HDS can exhibit dramatic sensitivity to initial conditions. For instance, if two switching surfaces intersect and the state is drawn from the initial condition towards the intersection, two completely different trajectories can result depending on which surface the state 'hits' in the proximity of the intersection. With all of these caveats in mind, effectively modeling HDS can be a daunting task. In order to simplify the modeling procedure, several computation tools have been developed including MATLAB/Simulink/Stateflow/SimEvents toolboxes, CAPE_OPEN, HYCON, gPROMS, Modelica, ABACUSS II, Omola/Omsim, Ptolemy II, and BaSiP.

The body of HDS modeling theory is gradually growing and only relatively recent (*c.* 1990's) efforts have attempted to unify HSD theory developed in different application areas. As such, the use of the aforementioned HDS models are only beginning to be used in civil engineering research. Before which, hybrid civil systems were typically analyzed through abstraction as continuous or discrete systems. Hybrid (cellular) automata have been used for modeling railway safety (Qin et al. 2009). Signal lights, the safety level, and speed limit where modeled as discrete states; while the train speed, wind speed, and rainfall encompassed the continuous states. Sewer systems, another type of transportation network, have also been modeled with hybrid cellular automata (Rohani and Hadi Afshar 2013). By combining continuous system dynamics and discrete event systems into a hybrid automata like model, Alzraiee, et al. modeled civil construction operations, e.g. earth moving (Alzraiee et al. 2012). Within air traffic alert and collision avoidance systems (TCAS), the aircraft position and speed are continuous variables, and thresholds and communication amongst aircraft are discrete. These were formalized into a hybrid automata and used to verify various properties of TCAS systems (Livadas et al. 2000). Plenty of opportunities exist for HDS models to be applied to model other civil systems. For instance, the later part of this chapter utilizes a hybrid automata for control of a hydronic pipe network.

5.1.3 Controller design for HDS

Extending the control theory of continuous dynamical systems to hybrid dynamical systems is a nontrivial undertaking. Closed form stability analysis is possible only for an over constrained subclass of problems as discussed in (Goebel et al. 2009). Additional challenges in controlling hybrid systems include identifying the discrete mode in which the system is operating, and nondeterministic mode switching leading to models necessitating differential inclusions instead of more simple differential equations. To this end, academics and industrial partners have developed sophisticated computational toolboxes which aid in the heavily iterative process of hybrid control system design (Lunze and Lamnabhi-Lagarrigue 2009).

Once the HDS model type and control architecture are chosen, challenges still await the hybrid control system designer: stability, robustness, existence, uniqueness, controllability, and computational complexity. As previously discussed, a switched system that switches between two stable sub-systems indefinitely can produce unstable behavior. If however the controller is designed to switch only a finite number of times, under certain assumptions, stability can be proven via “stability analysis through limited events” (Goebel et al. 2009). As with nonlinear continuous systems, Lyapunov stability theory and invariance properties are important tools for proving stability for a HDS. The idea of robustness in HDS is closely related to various definitions of stability, in that the system remains stable for arbitrary perturbations of state, control, and parameters; however, this sort of analysis may require conversion of the systems differential equations into more computationally complex differential inclusion (Goebel et al. 2009). In continuous nonlinear systems analysis it is typically assumed that the vector field $f(x)$ is Lipschitz-continuous, from which it can be proven that a unique solution exists for the differential equation $\dot{x}(t) = f\big(x(t)\big)$ for any initial $x(0)$. On the other hand, for HDS this Lipschitz condition does not often hold and the *well-posed-ness* of the solutions to the controlled system must be proven, i.e. a solution exists and is unique. (Lunze and Lamnabhi-Lagarrigue 2009). One important task of a control system designer is to determine if it is even possible to achieve the desired performance with a given input, also known as controllability and reachability analysis. Unfortunately, for HDS this is a complex endeavor. Even for relatively simple systems, it has be shown that no algorithm exists to decide in polynomial time if the system is controllable (Lunze and Lamnabhi-Lagarrigue 2009). The main hope is to leverage the specific structure of the system, for which an algorithm does exist, but in some cases the algorithm must still solve an NP-hard problem.

The available feedback control strategies for hybrid systems include the use of Lyapunov functions, linear matrix inequalities (LMI), and model predictive control (MPC). Lyapunov design methods can guarantee stability of the closed loop system, but selection of appropriate Lyapunov functions is nontrivial and highly problem dependent. Controller design by solving LMIs can lead to optimal controllers, but are currently only proven for a class of switched systems. Common among many of the software packages is the use of MPC as the preferred control strategy due to the easier nature of solving the optimal open loop control trajectory at each control step, instead of computing optimal closed loop control laws *a priori*.

Hybrid MPC has seen a lot of usage in process control problems with forms similar to the hydronic system case study used throughout this dissertation. This is due in large part to the complexity of HDS which prevent the derivation of closed-loop optimal control law. Chapter 3 presented the general form and objective of MPC, for continuous systems, which can be extended for control of HDS, most commonly using hybrid automata, MLD, or PWA HDS models. The MLD or PWA models can be created using the software HYSDEL, then imported into the Hybrid Systems Toolbox for MATLAB (Bemporad 2013). With the toolbox, the hybrid model-predictive control system can be simulated, tested, and implemented. If the computational effort required to solve the OL optimization problem online is too great, then the toolbox can also be used to generate an explicit MPC scheme for systems without time-varying disturbances. Explicit MPC segments the initial condition state-space and determines the optimal OL control for each segment by utilizing substantial off-line computing effort *a priori*. Even leveraging tricks such as explicit MPC, hybrid MPC is only applicable to a limited number of small systems with few binary variables (Lunze and Lamnabhi-Lagarrigue 2009).

As with hybrid modeling of civil infrastructure systems, application of hybrid control to these problems is still a topic of active research. Research which initially considered semi-active vehicle suspension systems using a MLD-MPC framework (Giorgetti et al. 2006), was extended to semi-active structural control problems (Elhaddad and Johnson 2013). The hybrid dynamics come to play in the semi-active control problem via the linear matrix inequality (LMI) passivity constraint of the actuator. That is, the actuator can only generate a force with the same sign as the actuator's differential velocity. Since MLD-MPC assumes a deterministic system and produces a deterministic response, and the earthquake disturbance is stochastic, a computationally expensive Monte-Carlo scheme was used off-line in order to generate an explicit MPC law. Hybrid dynamics can also arise in civil vibration control problem through faults in the physical or cyber system. Hybrid automaton have been used in this situation to account for the switching of the controller between 'healthy' and 'faulty' system states. Hybrid predictive control was achieve for multiple objectives in a public transit system using an evolutionary optimization approach with continuous dynamics linked to a discrete event system (DES). As previously mentioned regarding modeling, abstraction can be used in order to avoid the complexities of HDS, as was done with an on-off controlled HVAC system abstracted to a continuous control system (Aswani et al. 2012). Later in this chapter a hybrid automata is used in a hybrid MPC using a heuristic approach in order to reduce the computational effort in the wireless network on which the controller is implemented.

5.1.4 Control with hybrid automaton

Choosing between hybrid automaton, MLD, or PWA model architectures can be a difficult decision when designing a hybrid MPC system. Using a software like HYSDEL and MATLAB, a MLD or PWA model can be created that encapsulates all of the hybrid dynamics in a concise mathematical framework on which a MINLP can be executed to solve for the optimal OL control. However, if an attempt was made to implement such an architectures on a cyber-physical control system consisting of low-power wireless computation nodes, the computational complexity of running HYSDEL and solving the MINLP would likely be too great to achieve the desired real-time performance. (It could be possible to use the Hybrid Toolbox for MATLAB to generate an explicit MPC that could easily be run on a wireless network, but without HYSDEL and MATLAB embedded into the CPS, it would be prohibitively difficult with current technology to update the MPC with information on changes to the system.)

For HDS with relatively few discrete states, like some of the systems considered in this dissertation, hybrid automata can be an intuitive approach, especially for implementation into distributed code for calculating the MPC. Finite state machines, from which hybrid automata stem, are a common tool for computer scientists to schematically represent their computer programs. As such, using hybrid automata for MPC implemented in efficient code on low-power microcontrollers seems like an intuitive approach, even though some of the mathematical rigor of MLD-MPC is sometimes lost. (Not to say there isn't

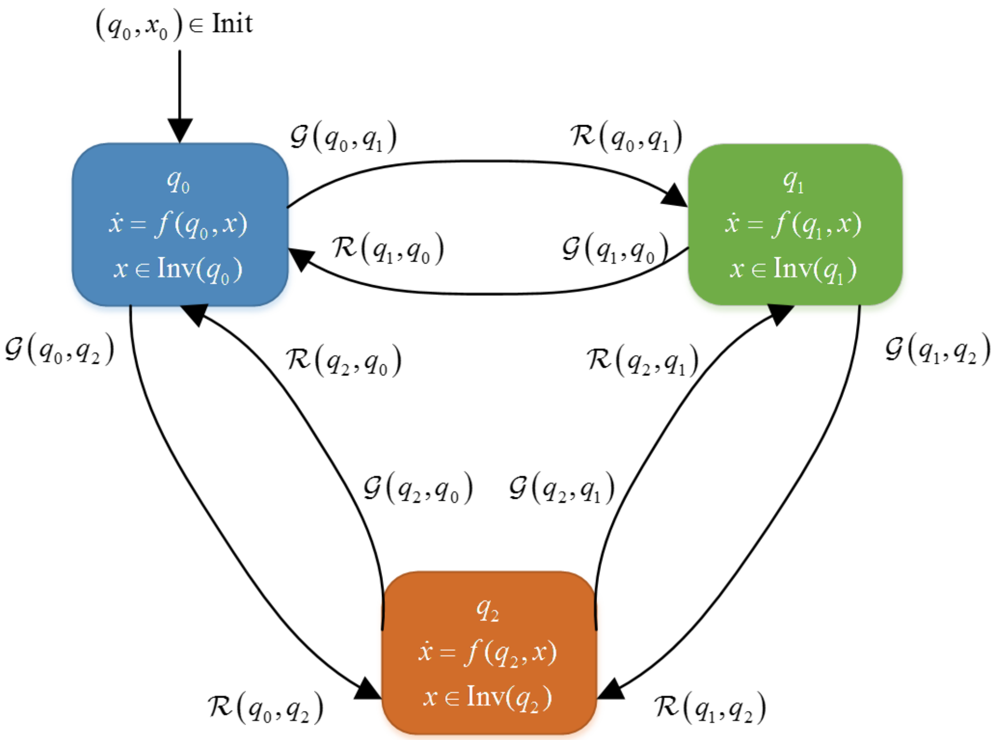


Figure 5.3 Generic hybrid automaton with three states adapted from (Lunze and Lamnabhi-Lagarrigue 2009)

a rigorous mathematical frame work for hybrid automata, see (Branicky 1998).) As seen in Section 4.3.2.3 the optimal control law for some MPC schemes, i.e. a minimum-time objective, can be solved in close form yielding the hybrid automata in Figure 4.4. A generic schematic of an uncontrolled hybrid automata with three discrete states $Q = \{q_0, q_1, q_2\}$, continuous vector space $\mathcal{X} \in \mathbb{R}^n$, and initial condition (q_0, x_0) is shown in Figure 5.3. The continuous state will continue to evolve according to $\dot{x} = f(q_0, x) \in Inv(q_0)$ until it satisfies either guard $\mathcal{G}(q_0, q_1)$ or $\mathcal{G}(q_0, q_2)$ and switches to state q_1 or q_2 respectively. During the switch, the continuous state may experience a jump according to the reset map $\mathcal{R}(\cdot,\cdot)$. Once the hybrid automata model has been created, the problems of stability, reachability, and the computing the optimal OL control trajectory from a given initial condition are to be addressed.

As already mentioned for HDS in general, determining stability of a hybrid automaton is a difficult problem. Various definitions of stability exist, along with different analysis methods for each: asymptotic stability, region stability, local asymptotic stability, and global uniform asymptotic stability. Methods exist for computing asymptotic stability for HDS in which the origin is an equilibrium point in the continuous state space for each of the discrete states in the hybrid automata. If the equilibrium differs within each subsystem, it may be helpful to prove region stability, wherein the system does not leave the desired region after a certain amount of time from the initial condition. Lyapunov's indirect method can be applied to linearized hybrid automata if all the discrete states are visited cyclically in order to determine local asymptotic stability. Global uniform asymptotic stability of hybrid automata made up of stable subsystems can be checked under certain conditions if the discrete transitions happen at a suitably slow rate, on average. Details on each of these methods for determining stability can be found in (Lunze and Lamnabhi-Lagarrigue 2009).

Similarly, reachability analysis of HDS can be challenging, and has been proven to be undecidable for hybrid automata in general (Lunze and Lamnabhi-Lagarrigue 2009). That is, iterative algorithms exist that can sometimes compute reachable domains, but they may not always converge. Most reachability algorithms rely on iterating through successor sets. The continuous successor set $\mathrm{Succ}_C(q, \mathcal{R})$ is the region of the continuous state space that is reachable from $\mathcal{R} \in \mathrm{Inv}(q)$ in mode q. This is not trivial to solve for, but a large body knowledge exists on continuous reachability analysis. The discrete successor set $\mathrm{Succ}_D(q, \mathcal{R})$ is the set of all discrete states that are reachable from q starting in $\mathcal{R} \in \mathrm{Inv}(q)$ after a single state transition. The hybrid successor set is then $\mathrm{Succ}_H(\mathcal{R}) = \mathrm{Succ}_C\big(\mathrm{Succ}_D(\mathcal{R})\big)$. When an exact designation of the reachable set is not necessary, e.g. when reachability is only used to verify safety limits, it may be easier to prove that a set is not reachable from an initial set, or that the system does not cross a particular boundary or 'barrier certificate'.

The method for implementing MPC using a hybrid automata varies depending upon the type of control, the objective, the number of states and modes in the HDS, the speed of the HDS dynamics, and the available computational power. If the hybrid automata has continuous control and is aiming to get into a closed target set, then Pontryagin's Maximum Principle can be extended in order to compute the necessary (but not necessarily sufficient) conditions for an OL control trajectory to be optimal. If the computing power is not available in real time, then explicit MPC may be a viable option. If discrete control points exist, possibly in addition to continuous actuation, solving for an optimal control yields a computationally complex MINLP, which grows with the length of the prediction horizon and number of discrete states. The computational burden can be reduced by implementing a heuristic which trades sub-optimality for improved computational speed. In the case of heuristic control, the heuristics is likely to be specific to the application and the form of the hybrid automata at hand. The next two sections of this chapter describe combining a hybrid automata based MPC with an agent-based control paradigm. This will entail utilizing a heurists based on a hierarchical control structure with agents in supply, demand, a system manipulation roles.

5.2 AGENT-BASED CONTROL (ABC)

Control of large-scale systems warrants a new approach extending the principles of MPC to a network of MPCs working together to control the subsystems comprising the plant. To this end, researchers have coined the field of distributed/decentralized model predictive control (MPC). Such systems come in a variety of architectures such as multiple isolated MPCs in each subsystem without direct communication with other coupled subsystems (i.e. *decentralized MPC*) and *distributed MPC* in which a subsystem's controlling agent consists of a local MPC with the ability to exchange information with neighboring agents to improve controller performance. The research proposed in this dissertation will focus on the latter case due to the increase in performance (Camacho and Bordons 1999) and the interest in studying wirelessly networked control systems.

Two classes of distributed model predictive controllers (DMPC) exist. Considered first are systems with substantial communication bandwidth which allow for the exchange of data several times throughout each control step. Under this communication architecture, agents can ensure that variables shared between neighboring subsystem models are equal for each agent. Convergence in shared variables under this type of DMPC results in control actions which are identical to those of an equivalent centralized MPC. Thus stability, robustness, and other properties known about centralized MPC are trivially extended to this class of DMPC. On the other hand, the throughput of a communication network may not allow for iterative data exchange during each control step, but instead only once after each agent has computed its local optimal control sequence. Under this communication architecture, data received from neighboring agents is delayed one control step and thus shared variables do not converge, and properties of centralized MPC are not as easily extended. Camponogara *et. al.* have however presented sufficient conditions under which closed-loop stability is guaranteed (Camponogara et al. 2002).

5.2.1 Challenges and opportunities

Agent-based control (ABC), a subclass of networked control, not only has networked controls' complications from date loss, latency, and rate limitation, but also its own problems with consensus, non-cooperation, and changing architecture. The hybrid nature of these systems makes verification a nontrivial, and often undecidable, task. However, ABC also builds on the advantages of networked control (e.g. configurability and large-scale) by bringing robustness through distribution, scalability, and ad-hoc reconfiguration. Utilizing wirelessly networked communication enables rapid low-cost deployment, mobile sensors and actuators, and computational ability collocated with sensors and actuators.

The delays from when a measurement is made to when it is processed, or from when a control is processed to when it is physically applied to the structure, define the system's latency. The latency arises from analog-to-digital conversion, computational delays within the nodes, network bandwidth limitation, and message collision or loss. For continuous systems with digital control, latency is a well-known obstacle that can cause destabilization if not properly accounted for. This behavior can be seen in oscillating systems in which the delay causes the control to be applied out of phase. Additionally for hybrid ABC, latency can cause delayed knowledge of changes in constraints which can cause infeasible controls to be calculated. As such, control agents must have fail-safe mechanisms in order to satisfy local and network constraints, regardless of delay or mis-information from other nodes. Similarly, consensus should be ensured in so much as possible between each node. Achieving consensus can be a hybrid control problem control problem in and of itself, e.g. consensus/synchronization of flashing in swarms of fireflies (Goebel et al. 2009).

Wireless agent-based control systems are made up of nodes, such as the *Martlet* (see Chapter 2). The concept of agents for wireless control comes logically from the collocated resources at each node: communication, computation, sensors and/or actuators, and the agents must utilize these resources to maximize their objective. Agents can increase their utility by requesting information from their neighbors, taking an action, or processing data. Communication between agents must be managed due to the limited bandwidth and reliability of wireless networks. Having no centralized point of failure means that ABC systems can be designed to be robust to cascading failures. One such way to do this is to link the communication network graph on top of the physical plants energy transfer graph, as was done in (Kane and Lynch 2012). This means that failures of nodes or links on the cyber-side result in only proportional failures in the physical plant. Achieving the robustness and deployability benefits of wireless ABC may come at the cost of sub-optimal performance since simplifying assumptions and heurists are often required in order to execute the complex algorithms in real-time. Verification of desired system performance and robustness for hybrid ABC systems can be undecidable, but sometimes possible through careful abstraction of the HDS (Benedetto 2008).

5.2.2 State of the art

Although the challenges of hybrid ABC control systems seem difficult to overcome, persistent researchers have made significant progress over the past two decades in realizing the benefits listed above. The general theory of distributed MPC is discussed in (Camponogara et al. 2002), while (Yin et al. 2000) addresses the problem of network scheduling, and associated solutions. Input constraints for distributed MPC can be handled by using the methods of (Liu et al. 2009). Optimization of bilinear systems, i.e. similar to those studied in Chapter 4, that are networked together (the physical processes, not necessarily the control agents) is a globally non-convex problem, but can be decomposed into convex sub-problems. The solutions to these sub-problems lead to upper and lower bounds on the global minimum (Floudas and Ciric 1989; Zamora and Grossmann 1998). Even when control problems are temporally distributed, the computation effort can be split amongst spatially distributed agents (Maestre et al. 2012).

The problem of ventilating deep mines exhibits hybrid dynamics, a changing environment, and mobile personnel that must have safe air to breath. Existing control strategies were simple (i.e. set fan to highest setting when area is occupied), yet inefficient due to a very conservative design. A wirelessly enabled controller based on a hybrid automaton was designed to control O_2, CO, and CO_2 level dynamics by efficiently switching the fan between high and low settings in order to adjust for the number of toxic-gas spewing trucks in each area of the mine. The safety limits of this hybrid controller were verified using a finite-state system abstraction of the hybrid automata (Witrant et al. 2010). A MPC was also analyzed and shown to outperform the hybrid switching controller, but execution was not possible in real-time. Additional civil infrastructure areas that have seen application of hybrid ABC include automated highways and air-traffic management (Lunze and Lamnabhi-Lagarrigue 2009). The next section of this chapter will study the control topics covered thus far: wireless networks, bilinear system control, model-predictive control, hybrid dynamics, and agent-based control applied to the shipboard chilled water plant described in Chapter 3.

5.3 Agent-based control of shipboard chilled water plant

The subject of this study is the hydronic cooling network of Chapter 3 in which the goal is to maintain the temperature of the blocks below a desired threshold by efficiently modulating the valves and pumps. It is a logical test-bed for implementing the wirelessly networked agent-based model-predictive controller alluded to above. The test bed features hybrid system dynamics, bilinear continuous dynamics, a networked architecture of the physical plant, and the desire for a modular and reconfigurable controller architecture.

5.3.1 Proposed controller

The *Martlet* and wireless architectures of Chapter 2, the test bed of Chapter 3, the bilinear system MPC of Chapter 4, and the hybrid- and agent-based control theory from earlier in this chapter culminate into the proposed control system. Although it is theoretically possible to design a comprehensive MPC for the hybrid system, limitations of wireless implementation prevent such a design from executing the MINLP in real-time, or in the case of explicit MPC, from adapting via changes to the MPC model. The centralized control objective for the hydronics test-bed, maintaining block temperature below a set-point by efficiently manipulating the pump speeds and valve configurations, was distributed through specific abstractions into a hierarchical ABC. The agent-based part of the controller is composed of a **pump agent**, a **valve agent**, **load agents** for each heater, and **thermal agents** for each block temperature to regulate. The hybrid-automata for the controller

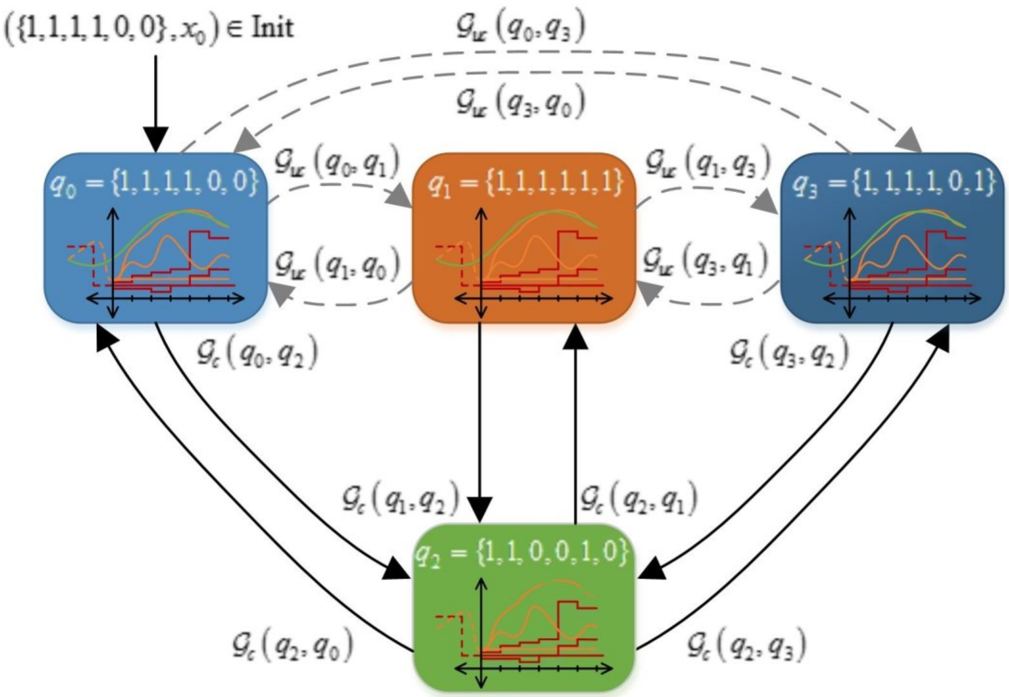

$(\{1,1,1,1,0,0\}, x_0) \in \text{Init}$
$\mathcal{G}_{uc}(q_0, q_3)$
$\mathcal{G}_{uc}(q_3, q_0)$
$q_0 = \{1,1,1,1,0,0\}$
$\mathcal{G}_{uc}(q_0, q_1)$
$q_1 = \{1,1,1,1,1,1\}$
$\mathcal{G}_{uc}(q_1, q_3)$
$q_3 = \{1,1,1,1,0,1\}$
$\mathcal{G}_{uc}(q_1, q_0)$
$\mathcal{G}_{uc}(q_3, q_1)$
$\mathcal{G}_{c}(q_0, q_2)$
$\mathcal{G}_{c}(q_3, q_2)$
$\mathcal{G}_{c}(q_1, q_2)$
$\mathcal{G}_{c}(q_2, q_1)$
$q_2 = \{1,1,0,0,1,0\}$
$\mathcal{G}_{c}(q_2, q_0)$
$\mathcal{G}_{c}(q_2, q_3)$

Figure 5.4 MPC + hybrid automata schematic of hydronic test bed

is shown in Figure 5.4. A continuous MPC controls the continuous pump speeds and block temperature transients. The discrete state vector $q = \{V1, V2, V3, V4, H1, H2\}$ describes if the respective valves and heaters are turned open/on, transitions according the uncontrolled guards $\mathcal{G}_{uc}$ and the controlled guards $\mathcal{G}_c$. Only four states exist out of the 2^6=64 possible because those are the only one determined to be optimal according to the control law below.

The demand side of the control problem was completely distributed amongst the thermal agents, while the supply side was just separated into two convex problems: pump control and valve reconfiguration. Under normal conditions on a ship it is expected that the hydronic cooling system operates near steady-state. This assumption justifies the splitting of the hybrid control problem into continuous transient control with the pumps and discrete steady-state control with the valves. The distributed bilinear MPC controls transients in block temperature (or even transients in pump costs if electrical demand increases due to loads in other systems of the ship). The valves switch

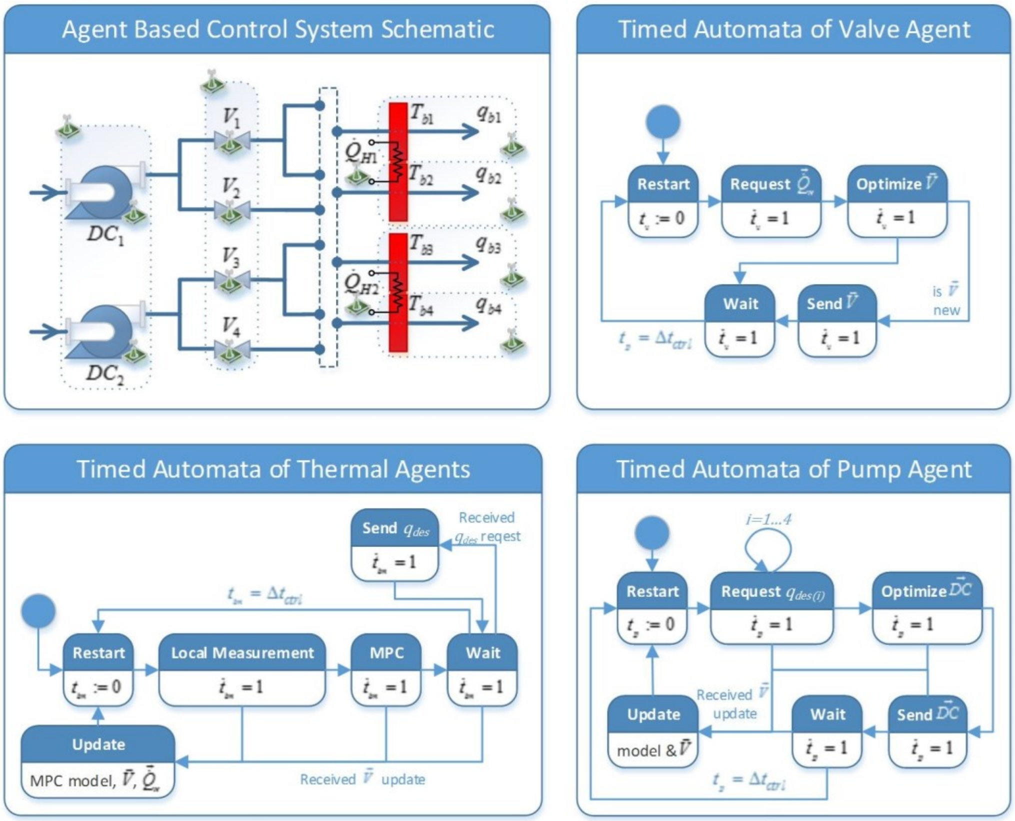


Figure 5.5 Agent-based control schematic and agent hybrid automata

to be in an optimal configuration for steady-state efficiency. Each thermal agent i is solving for their optimal OL flow trajectories. The first step of the trajectory is sent to the pump agent as a desired flow, q_{des_i}. The pump agent then determines the voltage, i.e. duty cycle, to apply to the pumps which satisfies each blocks desired flow. The entire control system (schematically depicted in Figure 5.5) can be thought of as a three step process: valve configuration, block OL control calculation, and pump satisfaction of flow demand, each process following its hybrid automata in Figure 5.5.

1) **Valve configuration:** The steady-state configuration of the pipe networks valves is a hybrid control problem with continuous states, the block temperature; continuous disturbances, the heat into the blocks (although for most the tests the heaters are only switched on and off); continuous controls, the pump speeds; and discrete controls, the valve configuration. Due to the valve-agent's assumption of operating in steady-state, the

dynamics can be abstracted resulting in an easier to solve MINLP. The program can be solved with a simple search across the 2^4=16 state-space of valve configurations, in which a convex optimization on the steady-state pump speed is performed in order to satisfy the steady-state flow demands of each block which are computed according to (5.7). This flow is what is required to maintain a steady-state temperature at or below the set-point for each block. See Section 3.2.3 for more details on the system model.

$$q_{ss_i} = \frac{1}{\alpha_1}\left(\max\left\{0, \frac{((T_a - T_{set})h_a + \dot{Q}_{H_i}}{T_w - T_{set}}\right\}\right)^{\frac{1}{\alpha_0}} \tag{5.7}$$

An exhaustive search of the discrete state space generates a switching rule for the valves that can be written entirely as a function of the heat coming from the systems loads, assuming the air temperature and water temperature are constant. This two dimensional switching map, represented by Figure 5.6, only uses 7 discrete states out of the 16 possible valve configurations. If the two heaters are only ever switched on or off, thereby increasing the possible discrete state-space by 2 factors of 2, the only valve configurations required are {1,1,0,0} and {1,1,1,1}. This leads to the 4-state hybrid-automata of Figure 5.3. The valve agent computes the optimal steady-state configuration at a fixed interval Δt_{ctrl} according to the timed automata in Figure 5.5. The process begins by requesting the heater agents' estimate of the heat input $\vec{\dot{Q}}_H$. If the valve agent determines that it is necessary to change the configuration, then wireless packets are broadcast to all of the units to inform the thermal and pump agents of the change in physical plant.

2) **Block OL control calculation**: Each block agent aims to receive enough flow to maintain its temperature at or below the set-point, while being respectful of the amount of power that would be required to generate such flow. The timed automata for the thermal agents in Figure 5.5 shows that every Δt_{ctrl} seconds they use their knowledge of the hydronic network's

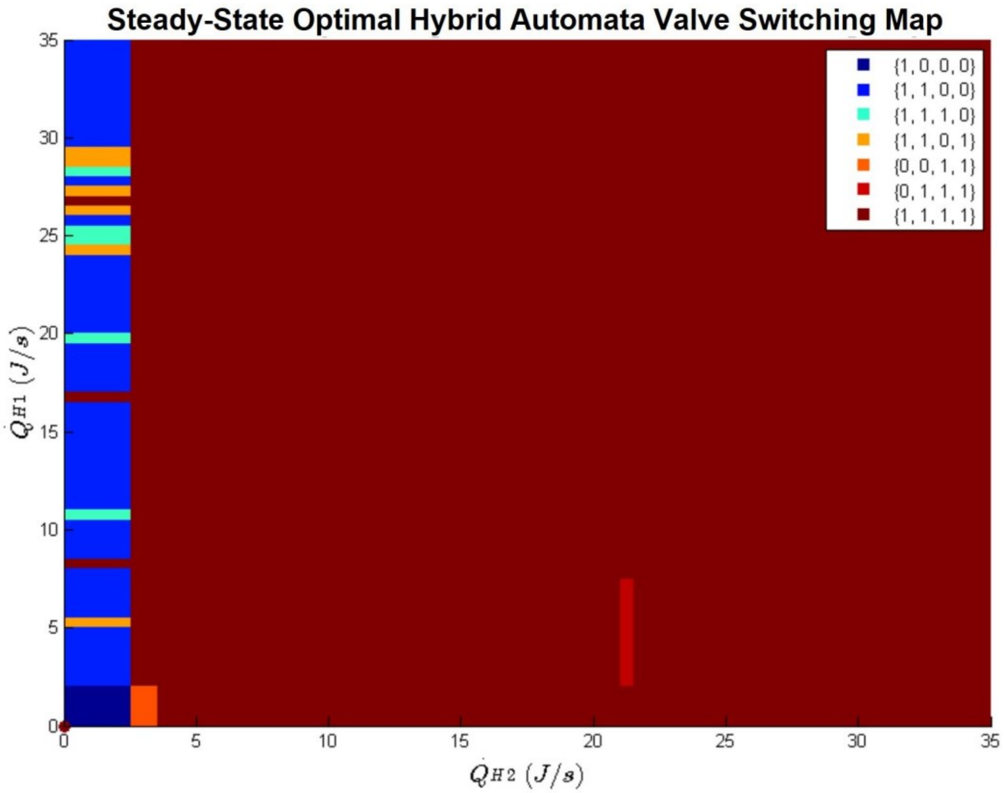


Figure 5.6 Hybrid automata valve switching map

configuration and their local temperature measurement as inputs into a MPC-type algorithm for generating a desired flow. The agents solve the open-loop bilinear-system control problem of Chapter 4 using the fast-MPC bilinear efficient cooling (BLEC) algorithm. They assume constant air temperature, water temperature, heat disturbance, and valve configuration over the prediction horizon, and neglect the heat transfer between adjacent blocks. The first step of the OL trajectory from the BLEC solution is then requested and received by the pump agent which completes the supply side of the MPC.

The model used for the MPC depends on the discrete state of the system, i.e. the valve configuration and heater status, which led to changes in the model parameters: maximum flow rate (and thus maximum heat transfer rate), and each block's flow-to-pump-power curves. To understand how the agents' calculate their utility, consider for example a simplified approximate model (a more advanced model fit was used for the

Table 5.1
Example flow and power allocation to thermal agents

	$\vec{V} = \{1,1,0,0\}$		$\vec{V} = \{1,1,1,1\}$	
	Flow $\left({}^{mL}/_{s}\right)$	Power (W)	Flow $\left({}^{mL}/_{s}\right)$	Power (W)
Pump 1	*20*	*24*	*20*	*24*
Pump 2	*0*	*0*	*20*	*24*
Block 1	$20 \cdot 25\% =$ *5*	$24 \cdot 25\% =$ *6*	$20 \cdot 25\% +$ $20 \cdot 25\% =$ *10*	$24 \cdot 25\% +$ $24 \cdot \frac{25\%}{2} =$ *9*
Block 2	$20 \cdot (25\% + 50\%) =$ *15*	$\mathbf{24} \cdot \left(25\% + \cdot \frac{50\%}{2}\right) =$ *12*	$20 \cdot (25\% + 50\%) =$ *15*	$\mathbf{24} \cdot \left(25\% + \cdot \frac{50\%}{2}\right) =$ *12*
Block 3	*0*	*0*	$20 \cdot (25\% + 50\%) =$ *15*	$\mathbf{24} \cdot \left(\frac{25\%}{2} + \cdot 50\%\right) =$ *15*
Block 4	$20 \cdot 50\% =$ *10*	$24 \cdot \left(\frac{50\%}{2}\right) =$ *6*	$20 \cdot 50\% +$ $20 \cdot 25\% =$ *15*	$24 \cdot \left(\frac{50\%}{2}\right) +$ $24 \cdot 25\% =$ *12*

experimentally implemented controller, see Section 3.2.3) in which flow is diverted 50%-50% at each bifurcation in the network, and the pumps' maximum flow and power is independent of valve configurations and fixed at 20 ${}^{mL}/_{s}$ and 24 $Watts$ respectively. Using the 50-50 split model and the network diagram in Figure 5.7, Table 5.1 was generated to show the flow allocated and the split of pump power assigned to each thermal agent. If block 1 were to request 10 ${}^{mL}/_{s}$ of flow in valve configuration {1,1,1,1}, it would assume it would 'pay a cost' of 9 W and blocks 2, 3, and 4 would each pay 12 W, 15 W, and 12 W respectively. However, if blocks 2, 3, and 4 request less than 15 ${}^{mL}/_{s}$ each, meaning they assume they consume less power than block 1 assumed they would, there will be an unaccounted for loss in utility across the system.

For now, this sub-optimality is neglected; however it could be accounted for in future iterations of the controller design if bandwidth and computational resources permitted arranging negotiations between thermal agents. Although the current design is not optimally energy efficient, it errors on the conservative side w.r.t. maintaining safe thermal

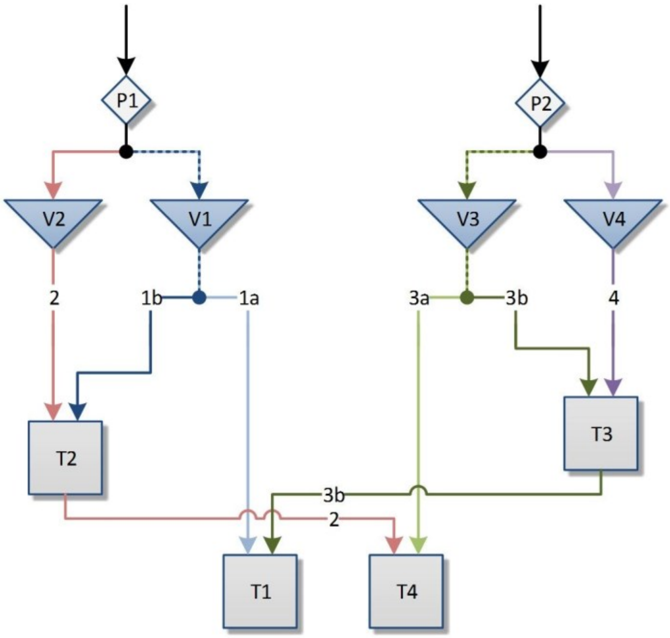


Figure 5.7 Hydronic network model used by thermal agents (neglecting inter-block thermal transfer)

limits. Three of the four thermal agents will often receive more flow then they requested, which will only improve their thermal performance. The MPC architecture updates the desired flow at each step to account for discrepancies in the expected flow and the neglected heat transfer between adjacent blocks.

3) **Pump satisfaction of flow demand**: After the valve agent sets the hybrid system's discrete state, and the thermal agents extract a desired flow from their calculated OL control trajectory, the pump agent is responsible for setting the speed at which to run the two pumps that satisfy the desired flow for each of the thermal agents. The optimal pumps speeds come from the solution to the problem (5.8). This minimization problem consists of a convex surface and LMI constraint which yields a convex program (Scherer and Weiland 2000) that can be solve efficiently. The objective function consists of a convex surface $\left|\vec{\mathcal{P}}(\vec{q})\right|$ mapping the $2x1$ vector of pump flows $\vec{q}$ to the norm of the power $\vec{\mathcal{P}}$ consumed by the pumps. The decision vector $\vec{q}$ is bound by the rectangular constraint from zero to maximum flow and the convex linear matrix inequality (LMI) constraint $A \cdot \vec{q} - \vec{q}_{des} \geq \vec{0}$. The

LMI constraint ensures that the $2x1$ vector of pump flows $\vec{q}$ generates a $4x1$ vector of flows to each block that is greater than the desired flows $\vec{q}_{des}$, according to the linear mapping A. Once a solution is found to the left hand side of (5.8) a simple algebraic mapping leads to the duty-cycles that the pumps should be set at. The program is solved using a bounded simplex algorithm (i.e. the Nelder-Meade method) with an additional weighted cost for violating the LMI constraint. The simplex algorithm is initialized with $\vec{q} = \vec{q}_{max}$, which is always admissible.

$$\left(\underset{\substack{\vec{q}\in[0,\vec{q}_{max}] \\ A\cdot\vec{q}-\vec{q}_{des}\geq\vec{0}}}{\operatorname{argmin}} \left|\vec{\mathcal{P}}(\vec{q})\right| \right) \rightarrow \overrightarrow{DC} \in [0,100] \tag{5.8}$$

The timed automata in Figure 5.5 for just the pump agent shows that every Δt_{ctrl} seconds, scheduled just after the thermal agents decide their desired flows, the pump agent requests the desired flow information for each thermal agent, computes the solution to (5.8), then commands the pumps to set the correct speed. If the valve agent informs the pump agent of a valve change, the pump agent must change the model used for optimization. The flow-to-power map, the maximum pump flow rates, the mapping from pump flow to block flows, and the mapping from pump flows to pump duty cycle are all functions of the discrete state of the hybrid automata Figure 5.4.

The coordination of these agents yields a control system that efficiently maintains safe block temperatures. The communication over the wireless network between agents occurs mainly using a scheduled TDMA arrangement, but allows for event based communication for intermittent events such as changes in heater output. Quality of service for this mixed TDMA and event-based scheme is ensured using a CCA scheme. The hybrid dynamics in the control system associated with latency and quantization within the wireless

Table 5.2
Times From 25°K Above, to Set-Point for Each Block in Each Discrete State

Discrete state vector	Δt_{b1}	Δt_{b2}	Δt_{b3}	Δt_{b4}
{1,1,0,0,1,0}	*38s*	*38s*	***192s***	*24s*
{1,1,1,1,1,0}	*25s*	*24s*	*20s*	*16s*
{1,1,1,1,0,1}	*22s*	*21s*	*23s*	*18s*
{1,1,1,1,1,1}	*25s*	*24s*	*23s*	*18s*

nodes is neglected due to their relatively minor role compared to the scale of the plant dynamics.

Reachability and worst case analysis of the hybrid control system are still open problems for theoretical analysis; however an empirical analysis can be undertaken showing expected worst-case performance bounds. In Chapter 4 it was observed that convergence of the BLEC algorithm occurred within 200 steps at an average of 1 step per second. Although the ABC in this chapter utilizes a fast-BLEC implementation, i.e. a solution was always found within the time-step even if sub-optimal, 200s iterations was used in this analysis as the worst-case reaction time. At typical air and water temperatures ($295^{\circ K}$), and maximum heat input ($35\ W$), it was empirically observed that the block temperature can rise up to $25^{\circ K}$ above the set-point ($312^{\circ K}$) within this 200s window. Table 5.2 shows the simulated time it took for each block to drop from $25^{\circ K}$ above the set-point to the set-point for the 4 discrete states in Figure 5.4 with pumps running at speed. The results shows that temperatures are brought to within safe limits in less than 38s for all cases, except for block 3 when pump 2 is not running. If such a case was expected in normal operation and was deemed unacceptable, an event-based trigger could change the discrete state out of the steady-state optimal configuration, and into a configuration to remove the undesirable transient.

In the following subsections, this wirelessly enabled agent-based model predictive controller will be shown to adequately control the response of the hydronics test bed and perform more effectively than other options considered.

5.3.2 Test scenarios

There are two main methods to benchmark controller designs: theoretically using closed form mathematical analysis, and empirically using experimental or simulated tests of typical and worst-case performance. When possible, closed form analysis is preferred, but is often not possible for complex control systems such as the one described herein. As such, an experiment was devised that tests both the steady-state and transient performance of the proposed controller. The experiment begins with blocks 3 and 4 at an elevated temperature of approximately $332^{\circ K}$, $20^{\circ K}$ above the set-point of $312^{\circ K}$. Once the blocks have been cooled, the heaters are switched on and off according to the schedule in Table 5.3, which also shows the optimal steady-state valve configuration for the proposed controller.

The proposed controller will be compared against a centralized 'bang-bang' controller, a centralized constant-control MPC, and an agent-based constant-control MPC. Since the compared controllers use separate objective functions, the fairest way to compare them is to calculate all the objective function values integrated over the entire time-history for each experiment. In

Table 5.3
Test Scenario Schedule for Heater, and ABC Valve State

Time (min)	Duration (min)	H1 (W)	H2 (W)	V1	V2	V3	V4
0	1	*0*	*0*	Open	Open	Open	Open
1	6	70	*0*	Open	Open	Closed	Closed
7	2	70	70	Open	Open	Open	Open
9	2	70	*0*	Open	Open	Closed	Closed
11	2	70	70	Open	Open	Open	Open
13	2	70	*0*	Open	Open	Closed	Closed
15	10	70	70	Open	Open	Open	Open
25	6	*0*	70	Open	Open	Open	Open
31	2	70	70	Open	Open	Open	Open
33	2	*0*	70	Open	Open	Open	Open
35	2	70	70	Open	Open	Open	Open
37	2	*0*	70	Open	Open	Open	Open
39	10	70	70	Open	Open	Open	Open
44	5	*0*	*0*	Open	Open	Open	Open

this way, the optimality of each controller for its objective can be analyzed along with how well it meets the objectives of other controllers. Additionally, total computation time of the MATLAB simulation executed in a single thread of a $3.2GHz$ Intel Core i5 CPU is used as a metric for comparing computational complexity of each controller.

Two different objectives are used for the benchmark controllers: a minimum time objective and a nonlinear utility maximization objective. The minimum-time benchmark controller simply checks every $\Delta t_{ctrl} = 5s$ if the block temperatures are above the $312^{\circ K}$ threshold. If so, it delivers the maximum possible flow to the noncompliant blocks until the temperature is once again safe. The utility maximization benchmark controllers, centralized and agent-based, optimize pump utility, thermal utility, and valve utility assuming a constant-control over their MPC horizon. The pump utility is computed using the same 3rd order polynomial fit of the flow-power curve for each pump as is used in the controller proposed herein. The thermal utility for each block is calculated using a *warning* temperature of $312^{\circ K}$ with quadratic violation cost and a *danger* temperature of $322^{\circ K}$ with quartic violation costs. The valve utility is calculated based on a cost associated with each time a valve state changes. The three utilities are then weighted with values of 0.95, 0.05, and 10^{-20} respectively. More details on the utility maximization benchmark centralized and agent-based controller algorithms and implementation can be found in (Kane and Lynch 2012).

5.3.3 Tuning and test results

As with nearly any control system design, the proposed wirelessly enabled model-predictive agent-based controller for hybrid-systems was designed and tuned in numerical simulation before ever being implemented on the physical plant. The simulation environment from Chapter 3 was used achieve the desired performance by adjusting the controller parameters: horizon length, horizon time-step, weighting balance between thermal and pump costs, and OL trajectory optimization algorithm and parameters. When

Table 5.4
Parameter Values Used to Test Proposed Controller

Parameter	Value	Parameter	Value
Pump simplex iterations	50	MPC horizon length	100 step
Control delay Δt_{ctrl}	5 s	MPC step length	5 s

porting code from the MATLAB simulation to the *Martlet*, optimization tolerances, such as for the TPBVP solver, were loosened to ensure real-time performance. Additionally, all calculations on the *Martlet* were executed in single precision (32-bit) floating point, while the MATLAB simulations used double precision (64-bit) floating point. Values selected for the parameters after tuning can be found in Table 5.4 and Table 4.1 for the general parameters and BLEC algorithm parameters respectively.

Once tuned, the proposed controller was compared against the already tuned benchmark controllers in simulation to test performance. This is an important step to validate that porting the controller form MATLAB simulations to C code running on the *Martlet* is worth the effort. In order to make the comparison between the different controllers as fair as possible each one uses the same set-point (i.e. $T_{set} = T_{max} = T_{warn} = 312^{\circ K}$) and each of the three objective functions were integrated over the simulation record for each of the four controllers tested. The results of this benchmark study are shown in Table 5.5 which was generated using the test scenario described in Section 5.3.2, modified slightly by removing the elevated initial condition of blocks 3 and 4, thus setting all blocks to $20^{\circ C}$. Since each controller is trying to minimize their respective objective, it would be expected that the bang-bang controller would have the lowest "$\int_{T_b > T_{set}} dt$" value, the constant control MPC (CCMPC) centralized and decentralized would have the same and lowest value for the "CCMPC Cost", and the prosed demand satisfaction agent-based MPC would have the lowest "BLEC Cost". This expectation only partially holds true. The bang-bang controller does do best at maintaining safe temperatures, with very little computation, but with the largest CCMPC and BLEC costs. Due to the sub-optimal constant control assumption and the approximate utility negotiation

Table 5.5
Benchmark Simulations' Objective Function Values

Controller Cost	$\int_{T_b>T_{set}} dt$	CCMPC Cost	BLEC Cost	CPU run time (s)
Bang-bang	**61 s**	**7.27**	**4.71**	**< 1**
CCMPC-Centralized	**3,717 s**	**0.45**	**1.79**	**458**
CCMPC-Agent	**209 s**	**1.64**	**1.17**	**10,487**
Demand satisfaction agent-based MPC	**1,670 s**	**0.42**	**0.16**	**6,940**

between agents, the centralized and agent-based CCMPC do not perform particularly well on any metric. As hypothesized, the proposed controller performs best on the 'BLEC Cost', yet also best on the 'CCMPC cost'. The proposed controlled has the second highest amount of time above the threshold, but that could be reduced with more tuning. The CPU run-time for the proposed controller is second highest, but will not be problematic for real-time performance because the computational effort is distributed across all the agents in the network.

After the efficacy of the proposed controller was shown in simulation compared to the benchmarks, it was deemed appropriate to port the code and embed the controller into a network of *Martlets*. The wireless control system was started from a command sent from a PC gateway, but from that point on the entire controller ran in the decentralized manner described. Each node saved its local measurements, observed packet losses, and computational results to its µSD card, creating log files of 2MB on average for each 49 minute experimental run. The data was then manually collected off the µSD cards and saved on a PC where post-processing was performed in MATLAB.

Figure 5.8 shows the post-processed time-history data from one experimental run of the proposed demand satisfaction agent-based model-predictive controller. The top four axes contain temperature (°C) information of the four blocks during the experiment. The thick black line plots the block temperature as measured at the beginning of each MPC period. The predicted open-loop *500s* long trajectory computed at each MPC period is plotted with colored dashed lines: dark blue being the first MPC period of the experiment, green being the MPC period at the middle of the experiment, and dark red being the last MPC period of the experiment. The $312^{\circ K}$ ($28.85^{\circ C}$) set-point is plotted with a black dashed line, and a green shading is applied to each block during each period that the heater is on and injecting 35 W of energy into each block. The bottom four axes contain information on the amount of flow $\left({}^{mL}/_{s}\right)$ supplied to each block. Similar to the temperature plots, the thick black line shows the amount of flow that was actually delivered to each block, the dashed colored lines show the predicted OL control trajectory of flow at each MPC period, and the greed shading signifies that the heater is applied to the block. Additionally, a gray line plots the flow demanded by each block.

From examination of Figure 5.8 it can be concluded that the proposed controller performed competently, but with room for improvement. Over the length of the experiment, packets were transmitted over the network with a success rate of *99.8%* on average, resulting in no observable loss in controller performance. The controller quickly realizes that blocks 3 and 4 need to be cooled, and generates enough flow to cool them nearly $20^{\circ K}$ within 50s. After which, the first two blocks begin to warm and flow is generated that meets, and matches closely, the desired flow of blocks 1 and 2 until they reach the set-point, with less than $1^{\circ K}$ of overshoot. A steady-state temperature nearly below the set-point is maintained until blocks 3 and 4 are heated up to the set-point. At which point an increased flow is provided to blocks 3 and 4, generating an excess flow to blocks 1 and 2. From there on out, blocks 3 and 4 are exclusively affecting the speed of the pumps, resulting in excess flow to blocks 1 and 2 maintaining their temperatures below the set-point. Poor performance is observed in block 3 during the latter $^{2}/_{3}$rds of the experiment. This is attributed to being uncontrolled when the system is in discrete state {1,1,0,0} and to excessive costs when in discrete state {1,1,1,1} as shown in Table 3.1 where block 3 has to ‘pay’ 15 W for the same 15 ${}^{mL}/_{s}$ for which blocks 2 and 4 are only ‘paying’ 12 W. Global sub-optimality is also caused by flow to blocks in excess of what was demanded. This is observed most notably in block two which is kept approximately 5$^{\circ K}$ below the set-point once blocks 3 and 4 begin to need cooling, resulting in approximately 3 $^{mL}/_{s}$ of excess flow.

5.4 Distributed control of hybrid systems conclusions

This chapter has extended the bilinear systems theory of Chapter 4 to applications with discrete and continuous controls, i.e. hybrid dynamical systems (HDS). A background on hybrid systems was provided showing their use across civil infrastructure, especially when implemented with computational and physical components. Multiple HDS model architectures for the sake of MPC were considered for embedding into a wireless CPS. Out of investigation of HDS models, the hybrid automata was found to fit best for infrastructure modeling, specifically for modeling the controllers of the hydronics test bed from Chapter 3. One key reason for this selection was the ability to distribute the computation load of a hybrid automata controller across a network of wireless agents in a hierarchical way. This led to a discussion on agent-based control systems, their challenges and opportunities, and why they are appropriate controller architectures for use in wireless cyber-physical control systems.

These control theories of BLS, MPC, HDS, and ABC were coalesced into a control system implemented on a network of *Martlets* aiming to control the entire hydronics test bed network. This led to the separation of the hybrid control problem in to a supervisory discrete valve switching controller for steady-state optimality and MPCs of transients using the continuous pump controls. The ABC consisted of a single valve agent controlling the discrete state and steady-state performance, four thermal agents maximizing their thermal utility while accounting for pump energy usage in generating their

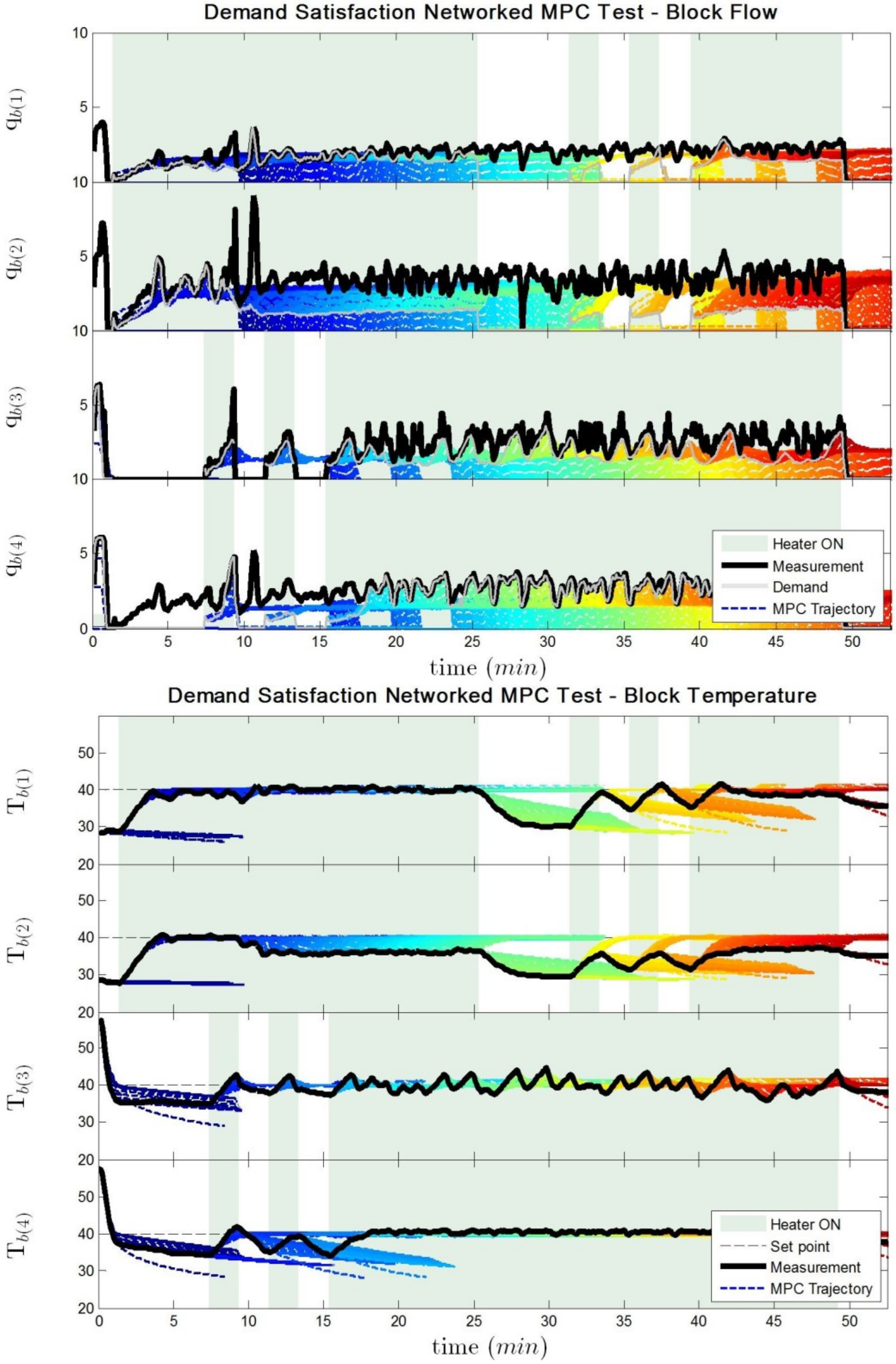


Figure 5.8 Demand satisfaction networked MPC experimental test results

desired OL control trajectories, and a single pump agent responsible for control of both supply-side pumps to ensure satisfaction of the flow demand of the thermal agents subject to the constraints of the pipe network in the current valve configuration. The proposed controller was then experimentally tested in a diverse testing scenario showing pleasing performance compared to benchmark controllers.

Chapter 6.

CONCLUSIONS AND FUTURE DIRECTIONS

This thesis built on and incorporated previous developments in wireless sensing and control of infrastructure, model-predictive control of bilinear systems, analysis and control of hybrid dynamical systems, and agent-based architectures for distributed control laws. These were investigated and developed with constraints and interdependencies associated with the cyber-physical systems viewpoint. The resulting proposed control system incorporated the new *Martlet* wireless hardware and a carefully selected bilinear system control approach which was distributed on a network of *Martlets*. Experimental results showed promise in comparison to established benchmark controllers. The hope is that these developments and proofs-of-concept systems will open the door for a new integrated cyber-physical systems approach to the challenges that civil infrastructure faces today and in the future.

6.1 SUMMARY OF RESULTS AND CONTRIBUTIONS

This thesis work is motivated by the grand challenge of improving civil infrastructure systems; wireless cyber-physical system (CPS) approaches can offer solutions to the vexing problems facing such systems including the need to enhance the resiliency of critical infrastructure systems. However, the application of cyber-technology to infrastructure improvement cannot be solely for the sake of intellectual interest; in order to generate meaningful momentum it must create a value proposition to infrastructure stakeholders. As CPS grow in scope, it is conceivable that hundreds of sensors and actuators will be at the cyber-physical interface. Reliance on centralized computing would likely not be scalable nor robust. As such, an opportunity exists to embed computing in a distributed manner into every practical location. Embedded computing has been enabled by exponentially improving wireless telemetry and embedded microcomputers. In these spatially distributed networked computers, sensing is available at almost every node; this is then utilized by collocated software agents in collaboration with agents at control nodes. All of the agents are programmed to act in a way that generates utility for themselves, which when performed in a collaborative manner also simultaneously achieves global design objectives.

The first chapter of this dissertation presented the prospect of applying recent technological developments within the CPS field to civil infrastructure management. An outline was given for the opportunity to build on previous work of wireless sensing and control systems to make infrastructure part of the CPS framework. A fundamental change in thinking must be made: motivation for wireless telemetry must go beyond viewing it as a direct replacement of wired links to viewing wirelessly networked microcontrollers as a more capable platform for other CPS function including computing. When using the metrics of traditional wired control and monitoring systems, the proposed architecture might not show dramatic improvements for all metrics. However, when viewing wireless monitoring and control subsystems of a CPS framework in terms of adaptability and robustness, such systems will far excel over wired counterparts. Hence, it is in applications where these metrics are high priorities that new wireless CPS can most excel and improve infrastructure system performance and resilience.

With sensors, actuators, and information processing embedded into computational agents deep within a physical plant of a CPS, a need arises for the nodes to communicate. In Chapter 2 a need is shown for a wireless computational node that researchers can use for developing this new category of wireless CPS. Combining developments for industrial, scientific, and consumer electronics to this end, the *Martlet* wireless sensing and control node was invented. Each *Martlet* node contains a dual-core microcontroller, a low-power wireless transceiver, extensive memory, and a modular extensible architecture that combine to bring a novel device to a market in need. The *Martlet* was first field tested in the desert of Los Alamos, New Mexico to monitor accelerations on a wind turbine tower for the purpose of structural performance and load monitoring. This first of many potential applications for the *Martlet* validated it as a tool capable of accurate data collection.

Chapter 3 details a developing cyber-physical system application which emerged from the U.S. Navy's new push for the integrated all-electric ship. Wireless technology has been shown to be a viable option for replacing wired links for cost effective retrofit of ship infrastructure. However, methods still need to be researched and developed for driving computational capabilities to be nearest to the 'leaf nodes' that exist at the lowest levels of the physical plant. This is especially necessary for combat vessels in order to achieve ship systems that are more robust and have greater 'fight through' capabilities.

The integrated hydronics, pipe network, and electrical sub-systems of the AES inspired the creation of a laboratory experiment set-up of a *Martlet* enabled CPS control architecture. This addressed the need for a test bed for benchmarking implementations of the new wireless CPS architectures. The test bed exhibits all of the characteristics of a CPS: multiple interdependent physical nonlinear and hybrid-dynamical processes, multiple points of control, competing objectives, multiple cybernetic processes, and many points of failure on both the physical and computational side of the larger CPS framework.

A step forward in implementing advanced control laws on embedded hardware is offered in Chapter 4. Two different mathematically equivalent models (i.e. linear model with state-dependent control constraints and a bilinear model with time-invariant rectangular constraints) were evaluated for their ability to control a class of nonlinear systems common in multiple civil infrastructure applications, including the hydronics test bed of Chapter 3. Analytical and experimental methods were used in the comparison to study how each would affect cyber-physical infrastructure design and performance, and ways that infrastructure constraints would affect model efficacy. The most fundamental hydronics system, reduced from the test bed in Chapter 3, was created to benchmark component level controllers developed in this chapter. In addition, multiple control objectives were investigated for their fitness in meeting actual control objectives for hydronic systems, such as those onboard ships, and also for their applicability to be embedded into the wireless sensing and actuation nodes of a CPS. An open-loop (OL) control trajectory solver was derived and simulated for each of the proposed objectives using both of the proposed models. Experimental data from the hydronics test bed of Chapter 3 showed that the proposed MPC with the bilinear model and explicit objective function was able to run suitably fast on the *Martlet* in real-time, realizing a controller that could replace existing suboptimal simple legacy control systems and the other models and objective functions studied. This laboratory experimental result also further validated the embedded computing capabilities of the *Martlet* and its framework for cyber-physical infrastructure.

Chapter 5 integrates the theories and developments discussed throughout the dissertation and sets forth an innovation in distributed advanced controllers of hybrid systems. This new control architecture was formed by bringing together developments in bilinear systems, model predictive control, hybrid-dynamical systems, and agent-based control. When compared with other architectures, hybrid-automata are presented as a practical framework for modeling infrastructure controllers with hybrid-systems, which contain continuous and discrete dynamics. A key to the success of the applied control architecture was a method of abstraction of steady-state and transient performance which decoupled the hybrid dynamics. Distributed control systems were demonstrated using two methods: holonomic segregation of system dynamics, and agent-based systems. The agent-based approach was shown to be more appropriate to wireless control systems, presumably because software agents could be collocated with the wireless control hardware. Laboratory implementation and experimental tests of the proposed wireless agent-based control framework for hybrid systems were conducted on the complete networked hydronics test bed of Chapter 3. The experimental results of the proposed control architecture, complete with this thesis' wireless platform and embedded bilinear MPC developments, showed pleasing performance compared to benchmark controllers.

6.2 FUTURE DIRECTIONS

The challenge to improve urban infrastructure cannot be overcome by any one discipline working in isolation; change will only come as experts in engineering, computer science, applied science, and social science work together to conceptualize and collaboratively address these problems. Chapter 1 described some of the obstacles inherent in moving away from traditional paradigms of control to a cyber-physical approach to the problem; these are depicted visually in Figure 1.2. By addressing some of these obstacles, this dissertation has moved forward the state-of-the-art in wireless control of cyber-physical infrastructure. However, as foreshadowed in Figure 1.2, there are areas that the dissertation does not directly address which hold the potential to further deepen the field and are worthy of investigation.

The *Martlet* was designed to meet the needs of wireless cyber-physical infrastructure researchers worldwide. As such, its design is extensible through application specific modular boards called *Martlet wings*. As the *Martlet* is interfaced to new infrastructure applications, these *wings* will need to be custom designed and fabricated following a set of loose standards already established. *Wings* for measuring acceleration and for the control of hydronic systems were created as part of this thesis, and Figure 2.8 through Figure 2.15 show the first round of *wings* created by all the researchers currently using the

Martlet platform. However, this only touches the tip of what is possible. Future efforts are needed to build out new *wing* boards designed to sense and control other systems.

With respect to the *Martlet*'s middleware, improvements need to be made to make it more power efficient, to increase the accuracy of network timing synchronization, and to implement multi-hop packet routing. Application specific software has yet to leverage one of the key design features of the *Martlet*, the secondary core, which is the control law accelerator; currently all that exists is a simple proof of concept. This illuminates the opportunity for research into the implementation of a control law onto the CLA, leaving processing power on the main CPU open for other computation tasks that are distributed through the network, such as model updating and system identification.

The embedded model predictive control (MPC) of bilinear systems developed in this dissertation used hydronics system as an application example, but the concept could be easily extended to many other infrastructure subsystems that exhibit bilinear behavior. It would be interesting to extend the analysis between linear and bilinear control model formulations that was done for hydronic systems to test whether the efficacy of the bilinear approach still holds for other application areas, such as forced-air building air conditioning systems or systems with faster dynamics such as semi-active structural control. Faster physical systems would likely require the development of more efficient algorithms for solving for optimal OL control trajectories. With regard to the specific hydronic test case used, extensions to the proposed controller could be made to either estimate the disturbance input without knowledge from a heater agent, or to utilize forecasts of the disturbance from the heater agent to improve MPC performance. The simple system used in Chapter 4 only contained a single continuous state; however it may be possible and more practical to extend the bilinear MPC to systems with multiple states, enabling control of larger subsystems before suboptimal distributed control abstraction is undertaken.

Hybrid dynamical systems (HDS) is a rapidly growing field, and this thesis only utilizes a small sampling of the developed theory in this emerging field. Hybrid automata are an attractive approach because they can be easily transferred into computer code, but they lack some of the rigor of other HDS models. Mixed logical-dynamic (MLD) systems with MPC seem to have the most robust methods of analysis, but such methods often employ computationally complex mixed-integer nonlinear programs (MINLP). Such an approach allows for continuous and discrete control to be considered together, instead of abstracted in a sub-optimal manner as was done in the proposed control architecture. If MLD systems are to be leveraged for wireless CPS, efficient algorithms for solving MINLP must be developed for execution on low-power microcontrollers, preferably solvable in a distributed manner. Regardless of other approaches, however, the hybrid-automata method develop herein could be improved in its own right. The valve switching strategy could be improved through thermal-event based switching to reduce the effect of extreme transients seen in Table 5.2. An explicit MPC could be implemented into the pump agent that maps desired flow of the thermal agents into a pump command for a given discrete system configuration. This would free the pump agents of their computational burden, thereby opening up opportunities of other computational tasks to be distributed throughout the network.

Agent-based systems are naturally suited for control of wireless CPS, but improvements can be made in the communication strategy between agents and implementation of multiple agents on a single wireless node. Communication between nodes could lead to better consensus among thermal agents, resulting in better estimates of energy usage and more optimal solutions. Additionally, communication could be used to inform agents of physical phenomena not modeled in the proposed controller, such as thermal transfer between adjacent blocks, or water temperature increases due to flow through upstream blocks. Agents need not be permanently associated with a particular node. Software agents could be designed to hop from node to node as computational resources are available throughout the network. For example, a wirelessly distributed simulated annealing model updating agent based architecture, like the one proposed in (Zimmerman and Lynch 2009), could be used to inform the entire network of changes to the physical plant. These updated plant models could then be used to update the MPC models, improving system performance and making it more robust.

Design of the integrated cyber-physical controllers, such as the final controller proposed in Chapter 5, is a complex task that requires analysis of the interdependencies made between the controller components (e.g. bilinear control plus hybrid control plus agent-based control), but also with other subsystems within the entire cyber-physical system being controlled. This research attempted to account for these interdependencies on a case-by-case basis using analytical and empirical methods in the most abstract way possible. However, the development of a unified framework of design guides, rules, and heuristics could prevent other CPS control system designs from undertaking such extensive design work. Existing computational design tools in the sub-fields of CPS control could be better brought into use by CPS researchers and extended to embedded wireless systems.

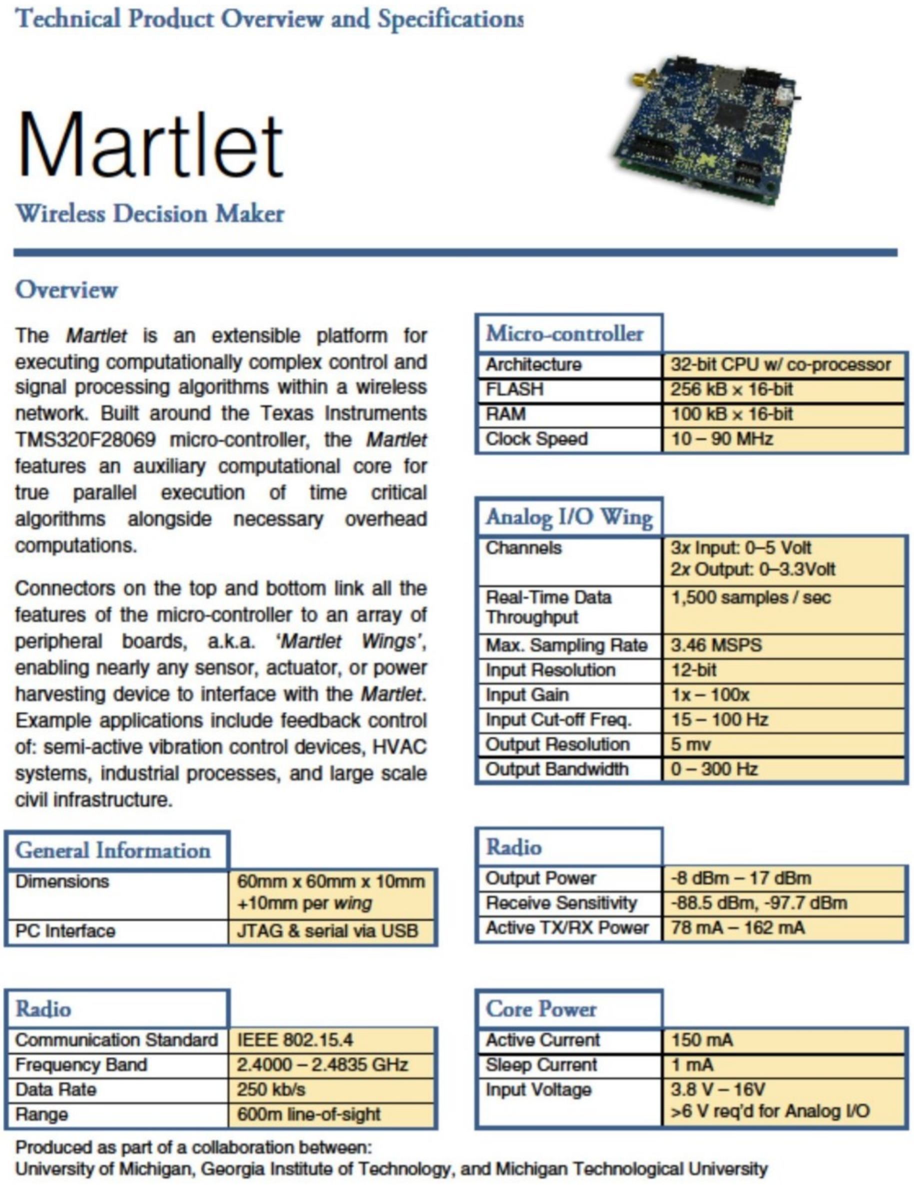
The wireless CPS topics in Figure 1.2 that were left untouched by this thesis were distributed system identification, integration of cloud-based computing, cyber-physical security, and user-interface and user experience improvements. Wirelessly distributed system identification could enable improved performance through adaptive model-predictive control, more cost effective maintenance from greater information about system health, and improved safety through early detection of probable catastrophic events. Cloud computing services are increasing in popularity among personal and business computational users, but cloud cyber-physical computing lags behind commercial development due to the careful balance required between security, the power of the cloud, and increased latency and wireless transmission power associated with data transfer to and from the cloud. Few wireless CPS exist without any human interaction, and many humans interact with consistent and important regularity. As such, the 'hard and cold computer system' could use a 'softer human touch' in the design process to make the human interaction more safe, effective, and pleasing. Human interaction also leads directly into the field of cyber-physical security. Security of cyber-physical systems is an interesting problem because vulnerabilities in the physical system can lead to computational consequences, and more importantly compromise of the computational systems could lead to real-world, potentially life threatening, consequences. For those interested in this topic, a study of early cyber-physical security work for the electric power grid can be found in (Sridhar et al. 2012).

If engineers continue to develop the wireless cyber-physical infrastructure, legislatures continue to understand its importance, and the population embraces it, then civilization can hope to live in a more convenient, safe, and sustainable world.

APPENDICES

Appendix A

MARTLET DATASHEET



Technical Product Overview and Specifications

Martlet

Wireless Decision Maker

Overview

The *Martlet* is an extensible platform for executing computationally complex control and signal processing algorithms within a wireless network. Built around the Texas Instruments TMS320F28069 micro-controller, the *Martlet* features an auxiliary computational core for true parallel execution of time critical algorithms alongside necessary overhead computations.

Connectors on the top and bottom link all the features of the micro-controller to an array of peripheral boards, a.k.a. *'Martlet Wings'*, enabling nearly any sensor, actuator, or power harvesting device to interface with the *Martlet*. Example applications include feedback control of: semi-active vibration control devices, HVAC systems, industrial processes, and large scale civil infrastructure.

General Information	
Dimensions	60mm x 60mm x 10mm +10mm per *wing*
PC Interface	JTAG & serial via USB

Radio	
Communication Standard	IEEE 802.15.4
Frequency Band	2.4000 – 2.4835 GHz
Data Rate	250 kb/s
Range	600m line-of-sight

Micro-controller	
Architecture	32-bit CPU w/ co-processor
FLASH	256 kB x 16-bit
RAM	100 kB x 16-bit
Clock Speed	10 – 90 MHz

Analog I/O Wing	
Channels	3*x* Input: 0–5 Volt 2*x* Output: 0–3.3Volt
Real-Time Data Throughput	1,500 samples / sec
Max. Sampling Rate	3.46 MSPS
Input Resolution	12-bit
Input Gain	1x – 100x
Input Cut-off Freq.	15 – 100 Hz
Output Resolution	5 mv
Output Bandwidth	0 – 300 Hz

Radio	
Output Power	-8 dBm – 17 dBm
Receive Sensitivity	-88.5 dBm, -97.7 dBm
Active TX/RX Power	78 mA – 162 mA

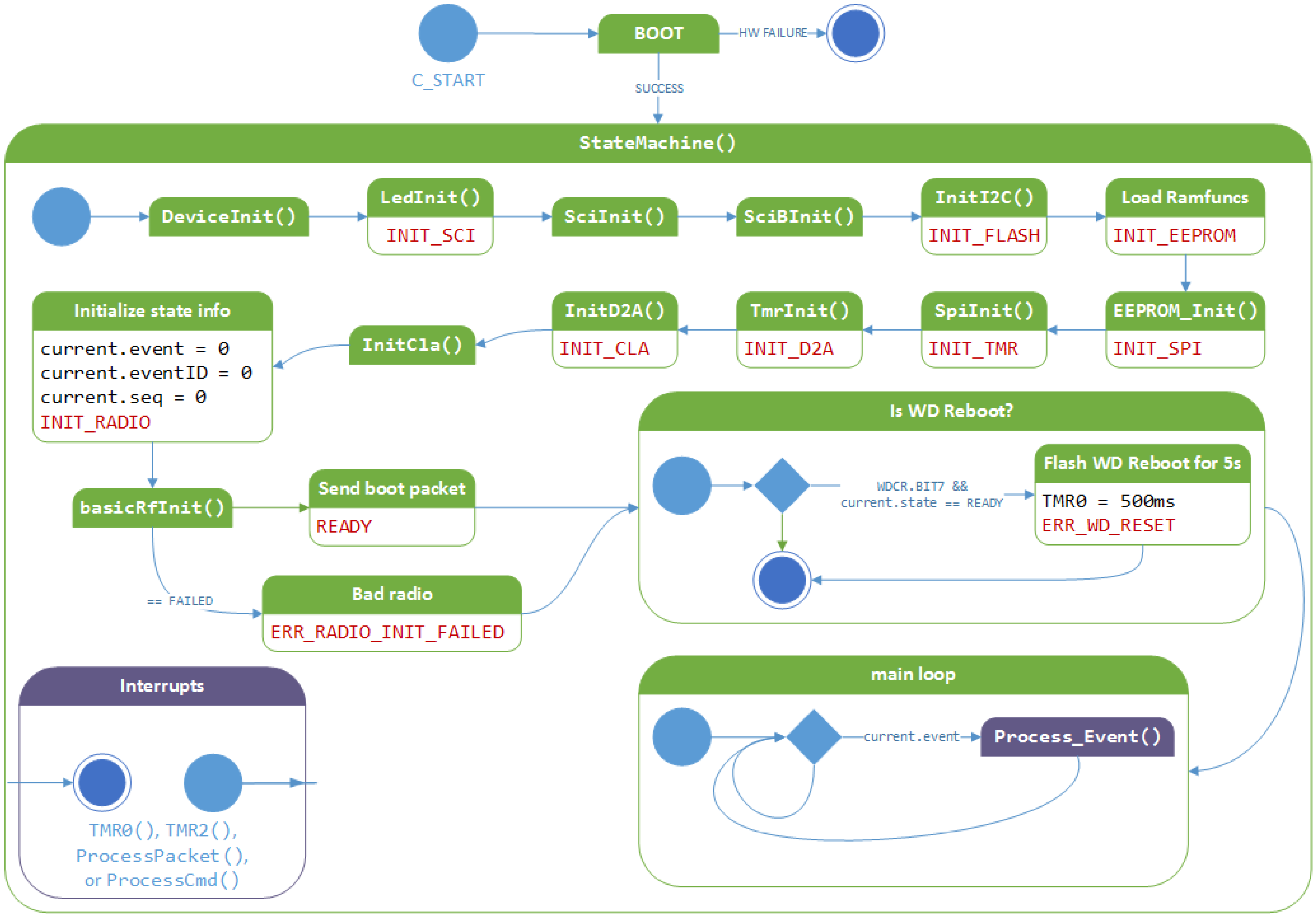
Core Power	
Active Current	150 mA
Sleep Current	1 mA
Input Voltage	3.8 V – 16V >6 V req'd for Analog I/O

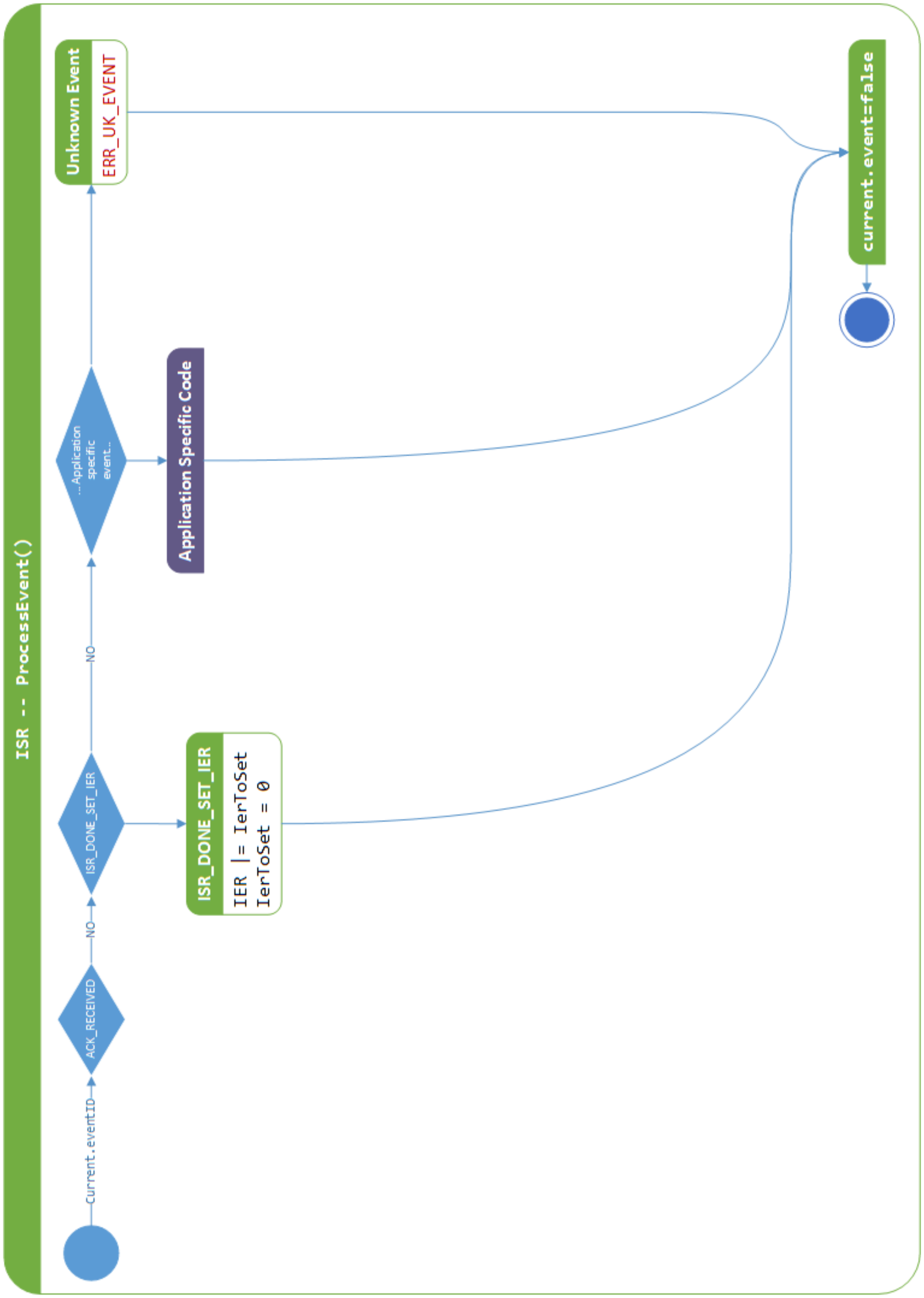
Produced as part of a collaboration between:
University of Michigan, Georgia Institute of Technology, and Michigan Technological University

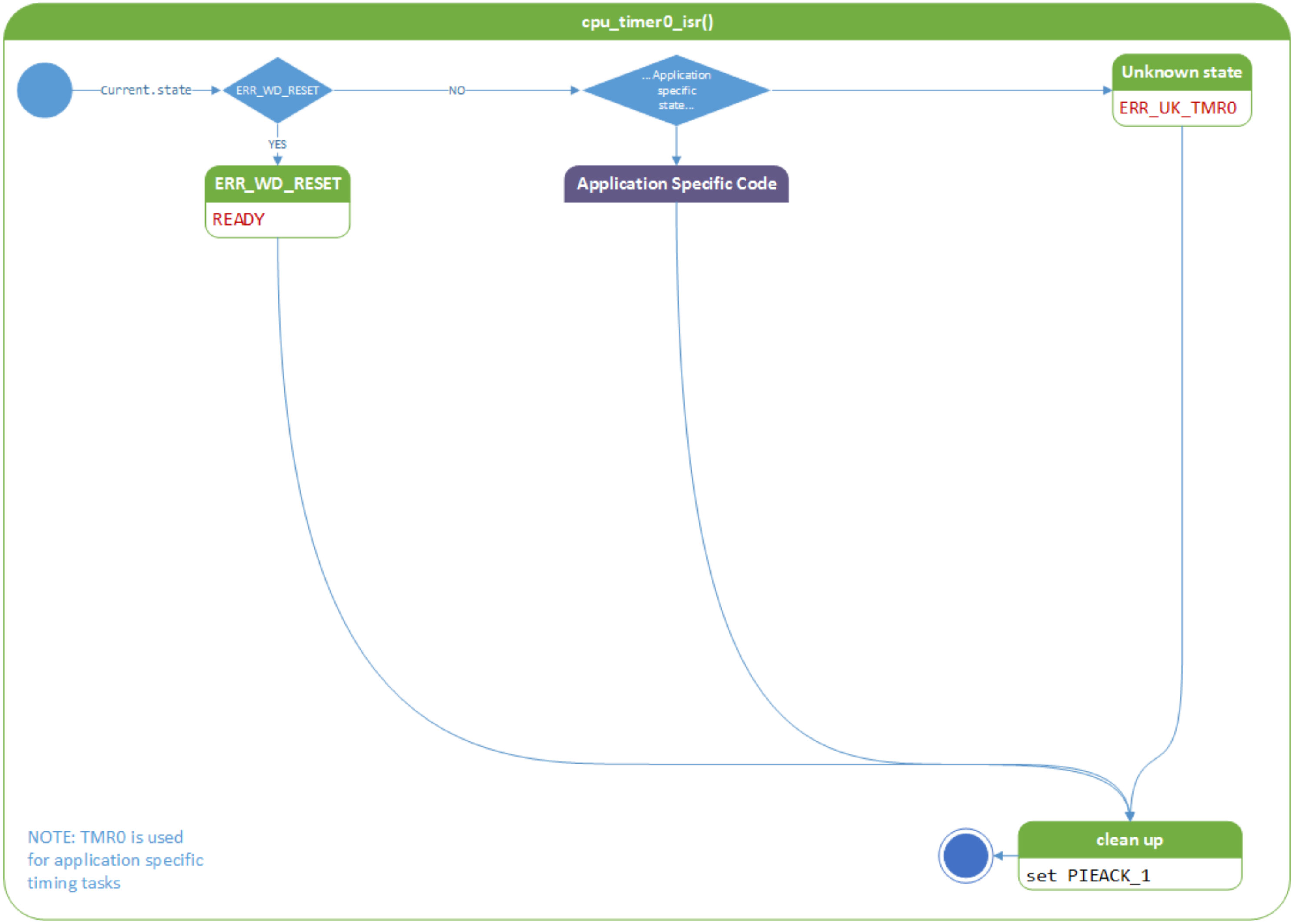
Appendix B

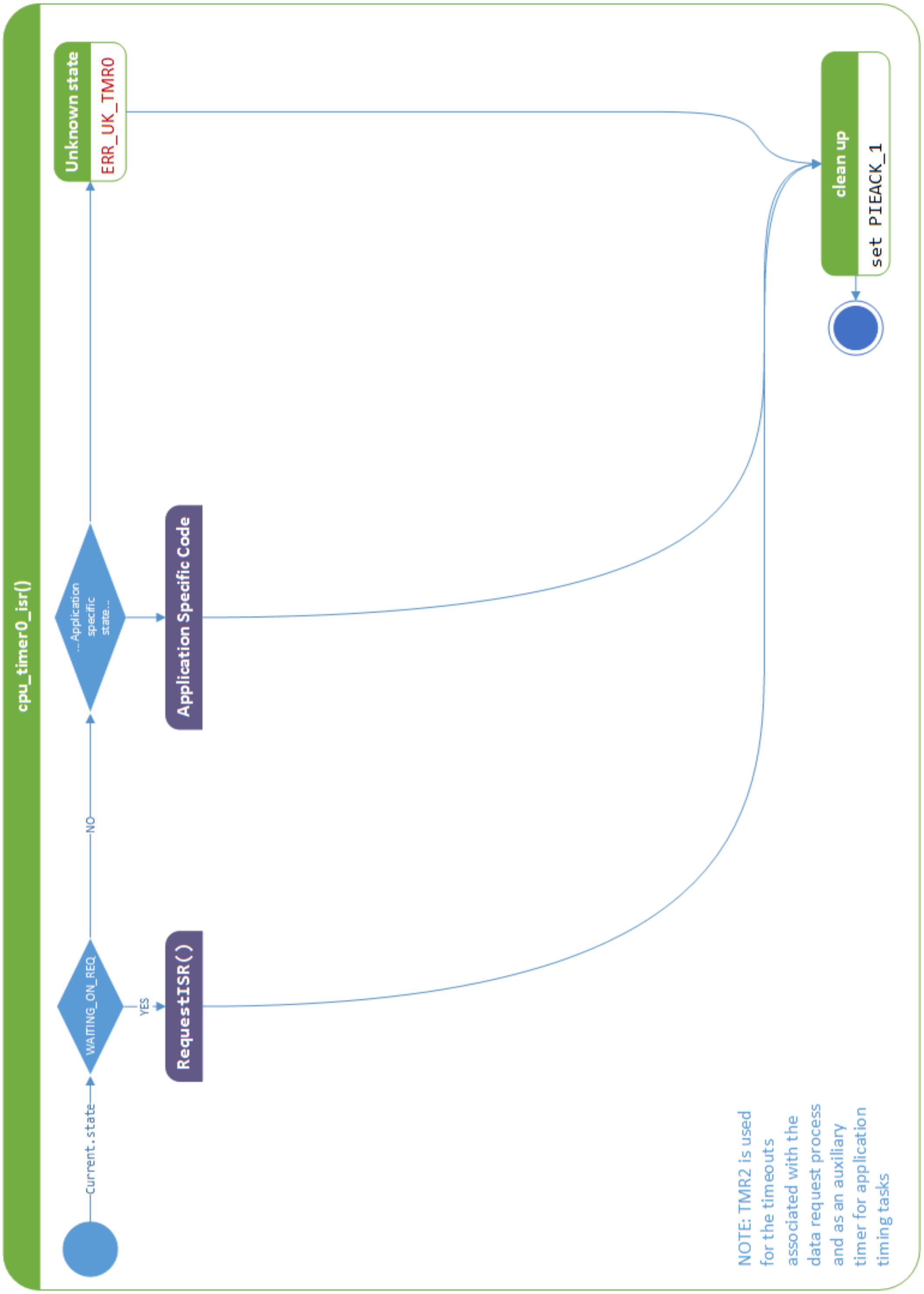
MARTLET STATE MACHINE

The *Martlet* achieves real-time operation with a software architecture defined by the state machine below. When powered up, or returning from a reset, the system starts at `C_START`, goes through the hardware `BOOT` process then enters the `StateMachine()` function. After initializing all of the software and peripherals, the *Martlet* sits and waits until an event is flagged, a packet is received, or an interrupt occurs, such as a timer reaching zero.

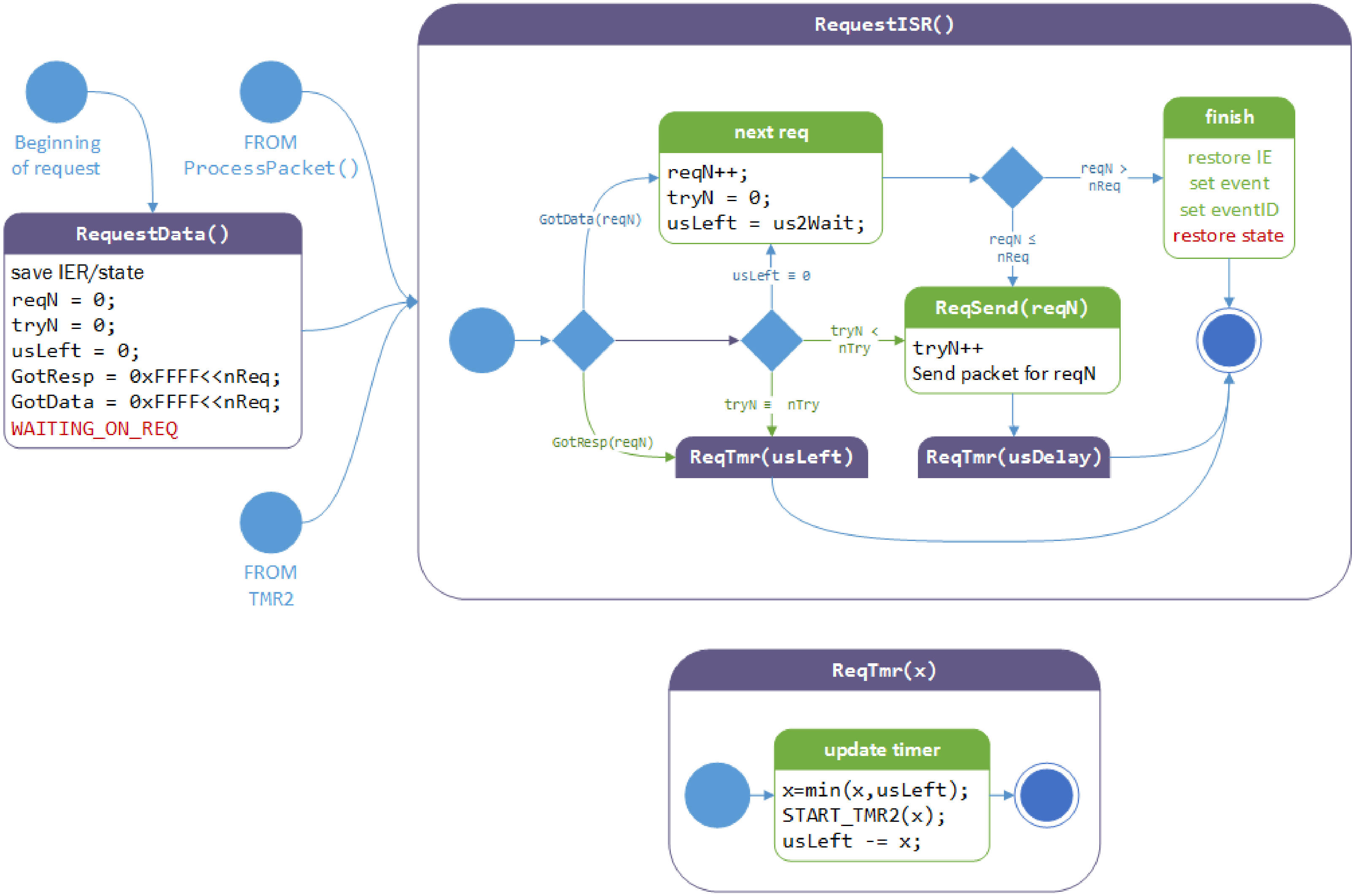

C_START
BOOT
HW FAILURE
SUCCESS
StateMachine()
DeviceInit()
LedInit()
INIT_SCI
SciInit()
SciBInit()
InitI2C()
INIT_FLASH
Load Ramfuncs
INIT_EEPROM
EEPROM_Init()
INIT_SPI
SpiInit()
INIT_TMR
TmrInit()
INIT_D2A
InitD2A()
INIT_CLA
InitCla()
Initialize state info
current.event = 0
current.eventID = 0
current.seq = 0
INIT_RADIO
basicRfInit()
Send boot packet
READY
== FAILED
Bad radio
ERR_RADIO_INIT_FAILED
Is WD Reboot?
WDCR.BIT7 &&
current.state == READY
Flash WD Reboot for 5s
TMR0 = 500ms
ERR_WD_RESET
main loop
current.event
Process_Event()
Interrupts
TMR0(), TMR2(),
ProcessPacket(),
or ProcessCmd()

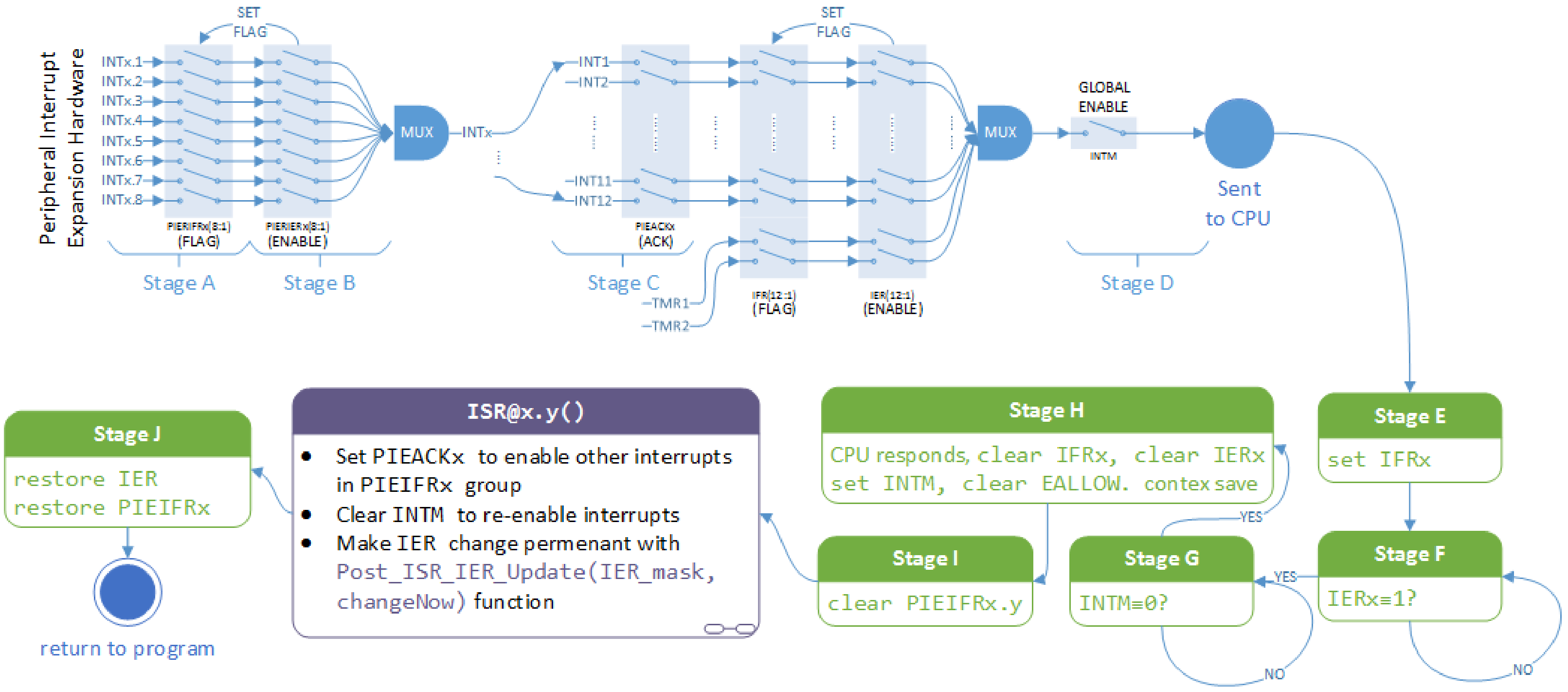

ISR -- ProcessEvent()
Current.eventID
ACK_RECEIVED
NO
ISR_DONE_SET_IER
NO
...Application specific event..
Application Specific Code
ISR_DONE_SET_IER
IER |= IerToSet
IerToSet = 0
Unknown Event
ERR_UK_EVENT
current.event=false


cpu_timer0_isr()
Current.state
ERR_WD_RESET
NO
...Application specific state...
Unknown state
ERR_UK_TMR0
YES
ERR_WD_RESET
READY
Application Specific Code
clean up
set PIEACK_1
NOTE: TMR0 is used for application specific timing tasks



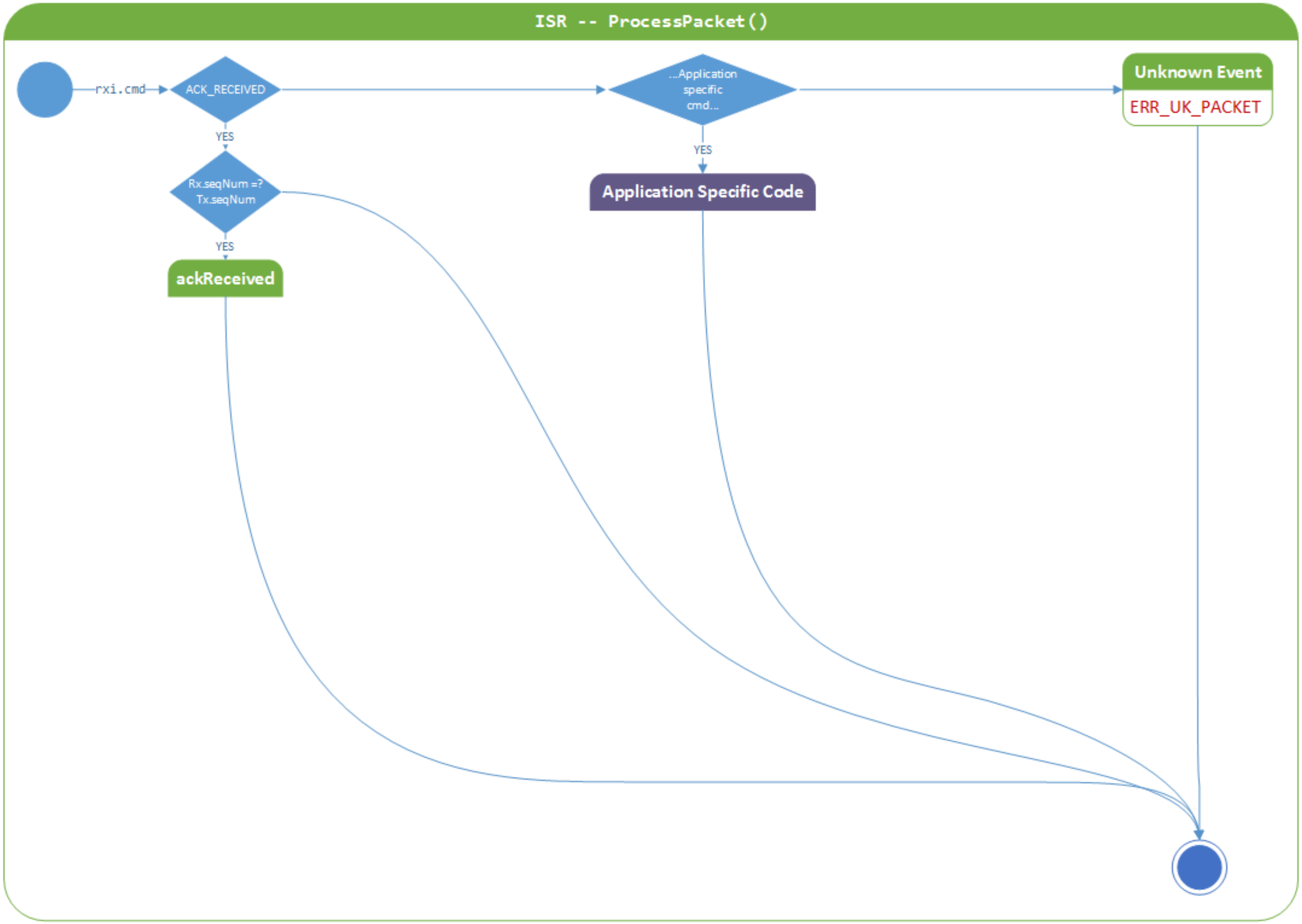
NOTE: TMR2 is used for the timeouts associated with the data request process and as an auxiliary timer for application timing tasks


Beginning of request
FROM ProcessPacket()
FROM TMR2
RequestData()
save IER/state
reqN = 0;
tryN = 0;
usLeft = 0;
GotResp = 0xFFFF<<nReq;
GotData = 0xFFFF<<nReq;
WAITING_ON_REQ
RequestISR()
GotData(reqN)
GotResp(reqN)
next req
reqN++;
tryN = 0;
usLeft = us2Wait;
usLeft ≡ 0
tryN < nTry
tryN ≡ nTry
reqN > nReq
reqN ≤ nReq
finish
restore IE
set event
set eventID
restore state
ReqSend(reqN)
tryN++
Send packet for reqN
ReqTmr(usLeft)
ReqTmr(usDelay)
ReqTmr(x)
update timer
x=min(x,usLeft);
START_TMR2(x);
usLeft -= x;



Peripheral Interrupt Expansion (PIE) vector table

	INT1.y	INT2.y	INT3.y	INT4.y	INT5.y	INT6.y	INT7.y	INT8.y	INT9.y	INT10.y	INT11.y	INT12.y
INTx.1	ADCINT1	EPWM1_TZINT	EPWM1_INT	ECAP1_INT	EQEP1_INT	SPIRXINTA	DINTCH1	I2CINT1A	SCIRXINTA	ADCINT1	CLA1_INT1	XINT3
INTx.2	ADCINT2	EPWM2_TZINT	EPWM2_INT	ECAP2_INT	EQEP2_INT	SPITXINTA	DINTCH2	I2CINT2A	SCITXINTA	ADCINT2	CLA1_INT2	-
INTx.3	-	EPWM3_TZINT	EPWM3_INT	ECAP3_INT	-	SPIRXINTB	DINTCH3	-	SCIRXINTB	ADCINT3	CLA1_INT3	-
INTx.4	XINT1	EPWM4_TZINT	EPWM4_INT	-	HRCAP3_INT	SPITXINTB	DINTCH4	-	SCITXINTB	ADCINT4	CLA1_INT4	-
INTx.5	XINT2	EPWM5_TZINT	EPWM5_INT	-	HRCAP4_INT	MRINTA	DINTCH5	-	ECAN0_INT0	ADCINT5	CLA1_INT5	-
INTx.6	ADCINT9	EPWM6_TZINT	EPWM6_INT	-	-	MXINTA	DINTCH6	-	ECAN1_INT1	ADCINT6	CLA1_INT6	-
INTx.7	TINT0	EPWM7_TZINT	EPWM7_INT	HRCAP1_INT	-	-	-	-	-	ADCINT7	CLA1_INT7	LVF
INTx.8	WAKEINT	EPWM8_TZINT	EPWM8_INT	HRCAP2_INT	USB0_INT	-	-	-	-	ADCINT8	CLA1_INT8	LUF


ISR -- ProcessPacket()
rxi.cmd
ACK_RECEIVED
YES
Rx.seqNum =? Tx.seqNum
YES
ackReceived
...Application specific cmd...
YES
Application Specific Code
Unknown Event
ERR_UK_PACKET

Appendix C

Minimum-Time Optimal Control Derivation

General minimum-time formulation

Given the system with state x, control u, and initial condition x_0

$$\dot{x}(t) = f(x,u,t) \quad x(t_0) = x_0 \tag{C.1}$$

Drive the system to a final state x_f in minimum time t_f subject to admissible controls $u(t) \in g(x,u,t)$

This can be described by the following minimization problem

$$\min_{\substack{\dot{x}(t)=f(x,u,t)\\ u(t)\in g(x,u,t)\\ x(t_0)=x_0\\ x(t_f)=x_f\\ t_0<t_f}} \bar{J}(x,u,t) \tag{C.2}$$

$$\bar{J} = \int_{t_0}^{t_f} dt \tag{C.3}$$

Replace the minimization's state constraint by augmenting the cost function $\bar{J}$ with a Lagrangian $p(t)$, yielding the new augmented cost function

$$J = \int_{t_0}^{t_f} [1 + p^T(t) f(x,u,t) - p^T(t)\dot{x}(t)] dt \tag{C.4}$$

The new optimization problem comes in mini-max form as follows

$$\min_{\substack{x(t)\\u(t)\in g(x,u,t)\\x(t_0)=x_0\\x(t_f)=x_f\\t_f\geq t_0}} \max_{p(t)} J(x,u,p,t) \tag{C.5}$$

Now start to compute the first-order necessary condition for optimality, $dJ = 0$, by applying the differential operator, δ, to both sides of J w.r.t. x, u, and t_f to each element in . [1]

- The first term $\int_{t_0}^{t_f}[1]dt = t_f - t_0$

$$\frac{\partial}{\partial x}\left(t_f - t_0\right)dx = 0; \quad \frac{\partial}{\partial u}\left(t_f - t_0\right)du = 0; \quad \frac{\partial}{\partial t_f}\left(t_f - t_0\right)dt_f = dt_f \tag{C.6}$$

- The second term $\int_{t_0}^{t_f}[p^T(t)f(x,u,t)]dt$

$$\begin{aligned}
&\frac{\partial}{\partial x}\int_{t_0}^{t_f} p^T(t)f(x,u,t)dtdx \\
&\qquad = \int_{t_0}^{t_f}(p_x^T f + p^T f_x)\delta x dt = \int_{t_0}^{t_f}(0 + p^T f_x)\delta x dt \\
&\qquad = \int_{t_0}^{t_f} p^T f_x \delta x dt
\end{aligned} \tag{C.7}$$

$$\frac{\partial}{\partial u}\int_{t_0}^{t_f} p^T(t)f(x,u,t)dtdu = \int_{t_0}^{t_f}(0 + p^T f_u)\delta u dt = \int_{t_0}^{t_f} p^T f_u \delta u dt$$

$$\frac{\partial}{\partial t_f}\int_{t_0}^{t_f} p^T(t)f(x,u,t)dt\delta t_f = 0$$

- The third term $\int_{t_0}^{t_f}[p^T(t)\dot{x}(t)]dt$

[1] From this point forward functions such as $f(x,u,t)$ may be abbreviated simply as f and their partial derivatives such as $\frac{\partial}{\partial x}f(x,u,t)$ may be abbreviated simply as f_x

$$\frac{\partial}{\partial x}\int_{t_0}^{t_f} p^T(t)\dot{x}(t)dtdx = \int_{t_0}^{t_f}[p^T\delta\dot{x}]dt$$
$$\frac{\partial}{\partial u}\int_{t_0}^{t_f} p^T(t)\dot{x}(t)dtdu = \int_{t_0}^{t_f}(0+0)\delta udt = 0 \tag{C.8}$$
$$\frac{\partial}{\partial t_f}\int_{t_0}^{t_f} p^T(t)\dot{x}(t)dtdt_f = 0$$

Combining each of the terms analyzed yields

$$dJ = dt_f + \int_{t_0}^{t_f}[(p^T f_x)\delta x + (p^T f_u)\ \delta u - p^T\delta\dot{x}]dt \tag{C.9}$$

Integrate by parts to get rid of the $\delta\dot{x}$

$$\begin{aligned} dJ &= dt_f + \int_{t_0}^{t_f}[(p^T f_x)\delta x + (p^T f_u)\delta u]dt \\ &\quad - \left([p^T\delta x]_{t=t_0}^{t=t_f} - \int_{t_0}^{t_f}\dot{p}^T\delta x dt\right) \\ dJ &= dt_f - [p^T\delta x]_{t=t_f} + [p^T\delta x]_{t=t_0} \\ &\quad + \int_{t_0}^{t_f}[(p^T f_x + \dot{p}^T)\delta x + (p^T f_u)\delta u\ ]dt \end{aligned} \tag{C.10}$$

If δx is interpreted as the variation in x "for time held fixed" (Bryson and Ho 1975), then

$$dx(t_f) = \delta x(t_f) + \dot{x}(t_f)dt_f \tag{C.11}$$

$$\delta x(t_f) = dx(t_f) - \dot{x}(t_f)dt_f \tag{C.12}$$

Updating dJ with this new definition yields

$$\begin{aligned} dJ &= dt_f - \left[p^T \cdot \left(dx - \dot{x}dt_f\right)\right]_{t=t_f} + [p^T \delta x]_{t=t_0} \\ &\qquad + \int_{t_0}^{t_f} [(p^T f_x + \dot{p}^T)\delta x + (p^T f_u)\delta u\,]dt \\ dJ &= \left[(1 + p^T \dot{x})dt_f - p^T dx\right]_{t=t_f} + [p^T \delta x]_{t=t_0} \\ &\qquad + \int_{t_0}^{t_f} [(p^T f_x + \dot{p}^T)\delta x + (p^T f_u)\delta u\,]dt \end{aligned} \tag{C.13}$$

Replace $\dot{x}(t_f)$ with the equivalent $f(x, u; t_f)$ and set $\delta x(t_0) = 0$ since $x(t_0)$ is given

$$\begin{aligned} dJ &= \left[(1 + p^T f)dt_f - p^T dx\right]_{t=t_f} \\ &\qquad + \int_{t_0}^{t_f} [(p^T f_x + \dot{p}^T)\delta x + (p^T f_u)\delta u\,]dt \end{aligned} \tag{C.14}$$

Next, choose the functions $p(t)$ to make the coefficients of $\delta x(t)$, and $dx(t_f)$ vanish

$$\dot{p}^T(t) = -p^T(t) f_x(x, u; t); \tag{C.15}$$

This choice of $p^T(t)$ leaves the following as the necessary first-order optimality condition

$$dJ = [(1 + p^T f)]_{t=t_f} dt_f + \int_{t_0}^{t_f} [(p^T f_u)\delta u]dt \tag{C.16}$$

By defining the Hamiltonian H, the necessary first-order optimality condition can be presented as

$$dJ = H(x, u, p; t_f)dt_f + \int_{t_0}^{t_f} H_u(x, u, p; t)\delta u dt \tag{C.17}$$

$$H(x, u, \lambda_0, p; t) = 1 + p^T(t) f(x, u, t) \tag{C.18}$$

Pontryagin's minimum principle states that the optimal state trajectory x^*, optimal control u^*, and corresponding Lagrange multiplier vector p^* must minimize the Hamiltonian H so that the following three conditions hold.

1) $H(x^*, u^*, p^*, t) \leq H(x^*, u, p^*, t)\ \forall\ t \in \left[t_0, t_f\right] and\ \forall\ u \in g(x, u, t)$

2) $H(x(t), u(t), p(t), t) = 0\ \forall\ t \in \left[t_0, t_f\right]$

3) $H_x = -\dot{p}^T = p^T f_x$

The Linear Formulation

Recall that the system dynamics can be described by

$$\dot{x}(t) = A\, x(t) + B\, u(t) + G \tag{C.19}$$

$$\begin{array}{ll} x(t) \coloneqq T_b(t) - T_w & u(t) \coloneqq \dot{Q}_w(t) = -x(t) h_w(t) \\ A \coloneqq -\dfrac{1}{mc_p} h_a < 0 & u(t)\left(u(t) - h_{w,max} x(t)\right) \leq 0 \\ B \coloneqq \dfrac{1}{mc_p} > 0 & G \coloneqq \dfrac{(T_a - T_w)\, h_a + \dot{Q}_h}{mc_p} \end{array} \tag{C.20}$$

The final state is defined as

$$x\left(t_f\right) = x_f = T_{set} - T_w \tag{C.21}$$

Since all the variables are scalar, scalar math can be used with these known bounds

$$A < 0 \tag{C.22}$$

The partial derivatives and the Hamiltonian then become

$$f_x = A; \quad f_u = 1 \tag{C.23}$$

$$H(x,u,p;t) = 1 + p(t)(Ax(t) + Bu(t) + G) \tag{C.24}$$

The three necessary conditions for optimality from the minimum principle become

1) $1 + p^* \cdot (Ax^* + Bu^* + G) \leq 1 + p^* \cdot (Ax^* + Bu + G)$

$$\forall\, t \in [t_0, t_f] \; and \; \forall\, u \in g(x,u,t)$$

Therefore $p^* = 0$ or $p^*u^* \leq p^*u \; \forall\, t \in [0, t_f] \; and \; \forall\, u \in g(x,u,t)$

$$\therefore u^* = \begin{cases} -x^* h_{w,max}, & p^* > 0 \quad x^* \geq 0 \\ 0, & p^* > 0 \quad x^* \leq 0 \\ 0, & p^* < 0 \quad x^* \geq 0 \\ -x^* h_{w,max}, & p^* < 0 \quad x^* \leq 0 \end{cases} \; \forall\, t \in [0, t_f] \tag{C.25}$$

2) $1 + p(t)(Ax(t) + Bu(t) + G) = 0 \; \forall\, t \in [t_0, t_f]$

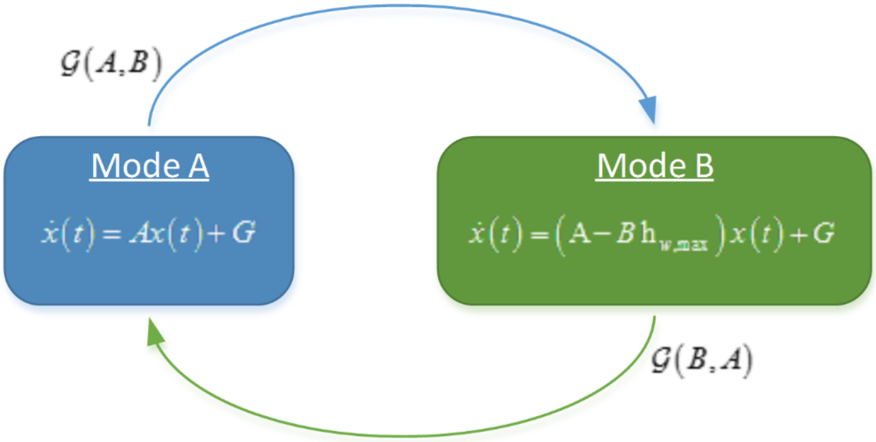
$$p(t) = \frac{-1}{Ax(t) + Bu(t) + G} \; \forall\, t \in [0, t_f] \tag{C.26}$$

3) $\dot{p}(t) = -Ap(t) \Rightarrow$

$$p(t) = p(t_f) e^{-(t - t_f)A} \tag{C.27}$$

4) $\therefore p(t)$ does not change sign from $t_0 \leq t \leq t_f$ and is independent of $u(t)$ and $x(t)$

Combining the state equation (C.19), the feedback function (C.25), and the knowledge that the sign of $p(t)$ does not change dynamically, it is possible to show that the system can be described by a hybrid automaton, i.e. a dynamic system with a finite number of operating modes which switch from one mode to another when the continuous state reaches a guard $\mathcal{G}(\cdot,\cdot)$. The evolution of the continuous states in a hybrid automaton is described by a different set of first-order differential equations in each operating mode. The switching between modes, and the associated continuous state evolution can be described by a hybrid automata.



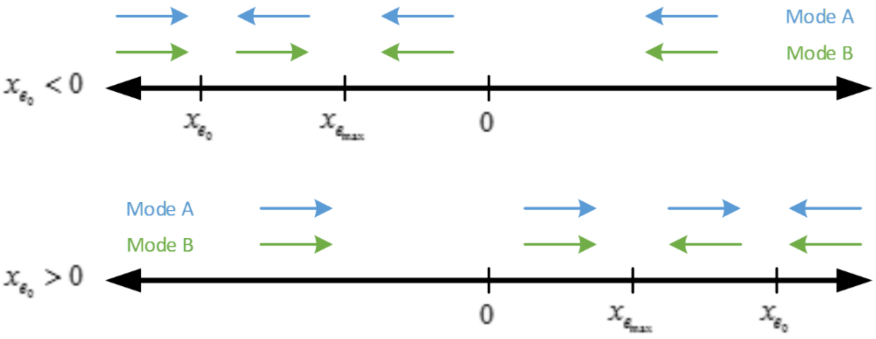
Studying the dynamics of mode A results in (C.28) and reveals that the state will monotonically approach x_{e_0}, where x_{e_0} is the uncontrolled equilibrium.

$$x(t) = (x_0 - x_{e_0})e^{(t-t_0)A} + x_{e_0} \tag{C.28}$$

Likewise, the dynamics of mode B in (C.29) reveal that the state monotonically approaches $x_{e_{max}}$, where $x_{e_{max}}$ is the equilibrium at maximum control.

$$x(t) = (x - x_{e_{max}})e^{(t-t_0)(A-Bh_{w,max})} + x_{e_{max}} \tag{C.29}$$

This direction of flow in the system with its dependence on operating mode and the values of x_{e_0} and $x_{e_{max}}$ is shown in the flow diagram recalling that $|x_{e_{max}}| < |x_{e_0}|$.



Because the controlled system is only capable of monotonically approaching x_{e_0} or $x_{e_{max}}$ not all final states x_f are reachable from initial states x_0. The inequalities in (C.30) show the unreachable sets that can be derived by analyzing the flow diagram.

$$\begin{array}{rr} x_f < x_0 < x_{e_0} < 0; & (x_{e_0} < 0) \wedge (x_f < x_{e_0}) \wedge (x_{e_{max}} < x_0) \\ (x_{e_0} < 0) \wedge (x_{e_{max}} < x_0 < x_f); & (x_{e_0} < 0) \wedge (x_0 < x_{e_0}) \wedge (x_{e_{max}} < x_f) \\ (0 < x_{e_0}) \wedge (x_f < x_0 < x_{e_{max}}); & (0 < x_{e_0}) \wedge (x_0 < x_{e_{max}}) \wedge (x_{e_0} < x_f) \\ (0 < x_{e_0}) \wedge (x_{e_0} < x_0 < x_f); & (0 < x_{e_0}) \wedge (x_f < x_{e_{max}}) \wedge (x_{e_0} < x_0) \end{array} \qquad \text{(C.30)}$$

The derivation of the system dynamics will consist of four parts: the configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that are not in (C.30) and x_0 and x_f are both positive; the configurations not in (C.30) and x_0 and x_f are both negative; the configurations not in (C.30) and $x_0 > 0$ and $x_f < 0$; and the configurations not in (C.30) and $x_0 < 0$ and $x_f > 0$.

First consider configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that are not in (C.30) and x_0 and x_f are positive:

> Assume that the system is operating in Mode A which from (C.25) implies $p(t) < 0$ and $u(t) = 0$. Under this assumption, the only permissible configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that drive the system from x_0 to x_f are (C.31) as seen in the flow diagram.
>
> $$\begin{array}{l} x_{e_0} < 0 < x_f < x_0 \\ 0 < x_0 < x_f < x_{e_0} \\ 0 < x_{e_0} < x_f < x_0 \end{array} \qquad \text{(C.31)}$$
>
> Since the value of $u(t)$ and the sign of $p(t)$ were implied, the second condition of the minimum principle (C.32) can be used to validate the assumption. Combining the valid conditions from (C.31) with the invalid conditions from (C.32) yields the stricter set of valid conditions (C.33). Therefore the assumption that the system is operating in Mode A is only valid for (C.33).

$$\begin{aligned} p(t) = \frac{-\lambda_0}{Ax(t) + G} < 0 \; \forall \; t \in \left[t_0, t_f\right] \\ \Leftrightarrow Ax(t) + G > 0 \; \forall \; t \in \left[t_0, t_f\right] \\ \Leftrightarrow x(t) < x_{e_0} \; \forall \; t \in \left[t_0, t_f\right] \\ \Leftrightarrow \left(x_0 < x_{e_0}\right) \wedge \left(x_f < x_{e_0}\right) \end{aligned} \tag{C.32}$$

$$0 < x_0 < x_f < x_{e_0} \tag{C.33}$$

Next, assume that the system is operating in Mode B which from (C.25) implies $p(t) > 0$ and $u(t) = -x(t)h_{w,max}$. Under this assumption, the only permissible configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that drive the system from x_0 to x_f are (C.34) as seen in the flow diagram.

$$\begin{aligned} & x_{e_{max}} < 0 < x_f < x_0 \\ & 0 < x_0 < x_f < x_{e_{max}} \\ & 0 < x_{e_{max}} < x_f < x_0 \end{aligned} \tag{C.34}$$

Since the value of $u(t)$ and the sign of $p(t)$ were implied, the second condition of the minimum principle (C.35) can be used to validate the assumption. Combining the valid conditions from (C.34) with the invalid conditions from (C.35) yields the stricter set of valid conditions (C.36). Therefore the assumption that the system is operating in Mode B is only valid for (C.36).

$$\begin{aligned} p(t) = \frac{-\lambda_0}{\left(A - Bh_{w,max}\right)x(t) + G} > 0 \; \forall \; t \in \left[t_0, t_f\right] \\ \Leftrightarrow \left(A - Bh_{w,max}\right)x(t) + G < 0 \; \forall \; t \in \left[t_0, t_f\right] \\ \Leftrightarrow x(t) > x_{e_{max}} \; \forall \; t \in \left[t_0, t_f\right] \\ \Leftrightarrow \left(x_{e_{max}} < x_0\right) \wedge \left(x_{e_{max}} < x_f\right) \end{aligned} \tag{C.35}$$

$$\begin{aligned} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \end{aligned} \tag{C.36}$$

In summary, since (C.33) and (C.36) do not overlap, the operating mode can be determined *a priori*.

$$x(t) = \begin{cases} \left(x_0 - x_{e_0}\right)e^{(t-t_0)A} + x_{e_0}; & \begin{cases} 0 < x_0 < x_f < x_{e_0} \\ \dots \end{cases} \\ \left(x_0 - x_{e_{max}}\right)e^{(t-t_0)\left(A-Bh_{w,max}\right)} + x_{e_{max}}; & \begin{cases} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \\ \dots \end{cases} \\ \dots; & \dots \end{cases} \tag{C.37}$$

Next consider configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that are not in (C.30) and x_0 and x_f are negative:

Assume that the system is operating in Mode A which from (C.25) implies $p(t) > 0$ and $u(t) = 0$. Under this assumption, the only permissible configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that drive the system from x_0 to x_f are (C.38) as seen in the flow diagram.

$$\begin{aligned} x_0 < x_f < x_{e_0} < 0 \\ x_{e_0} < x_f < x_0 < 0 \\ x_0 < x_f < 0 < x_{e_0} \end{aligned} \tag{C.38}$$

Since the value of $u(t)$ and the sign of $p(t)$ were implied, the second condition of the minimum principle (C.39) can be used to validate the assumption. Combining the valid conditions from (C.36) with the invalid conditions from (C.39) yields the stricter set of valid conditions (C.40). Therefore the assumption that the system is operating in Mode A is only valid for (C.40).

$$\begin{aligned} p(t) = \frac{-\lambda_0}{Ax(t) + G} > 0 \; \forall \, t \in \left[t_0, t_f\right] \\ \Leftrightarrow Ax(t) + G < 0 \; \forall \, t \in \left[t_0, t_f\right] \\ \Leftrightarrow x(t) > x_{e_0} \forall \, t \in \left[t_0, t_f\right] \\ \Leftrightarrow \left(x_{e_0} < x_0\right) \wedge \left(x_{e_0} < x_f\right) \end{aligned} \tag{C.39}$$

$$x_{e_0} < x_f < x_0 < 0 \tag{C.40}$$

Next, assume that the system is operating in Mode B which from (C.25) implies $p(t) < 0$ and $u(t) = -x(t)h_{w,max}$. Under this assumption, the only permissible configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f that drive the system from x_0 to x_f are (C.41) as seen in the flow diagram.

$$\begin{gathered} x_0 < x_f < x_{e_{max}} < 0 \\ x_{e_{max}} < x_f < x_0 < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{gathered} \tag{C.41}$$

Since the value of $u(t)$ and the sign of $p(t)$ were implied, the second condition of the minimum principle (C.42) can be used to validate the assumption. Combining the valid conditions from (C.41) with the invalid conditions from (C.42) yields the stricter set of valid conditions (C.43). Therefore the assumption that system is operating in Mode B is only valid for (C.43).

$$\begin{aligned} p(t) = \frac{-\lambda_0}{\left(A - Bh_{w,max}\right)x(t) + G} < 0 \; \forall \, t \in \left[t_0, t_f\right] \\ \Leftrightarrow \left(A - Bh_{w,max}\right)x(t) + G > 0 \; \forall \, t \in \left[t_0, t_f\right] \\ \Leftrightarrow x(t) < x_{e_{max}} \; \forall \, t \in \left[t_0, t_f\right] \\ \Leftrightarrow \left(x_0 < x_{e_{max}}\right) \wedge \left(x_f < x_{e_{max}}\right) \end{aligned} \tag{C.42}$$

$$\begin{gathered} x_0 < x_f < x_{e_{max}} < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{gathered} \tag{C.43}$$

In summary, since (C.33), (C.36), (C.40), and (C.43) do not overlap, the operating mode can be determined *a priori*.

$$x(t) = \begin{cases} \left(x_0 - x_{e_0}\right)e^{(t-t_0)A} + x_{e_0}; & \begin{cases} 0 < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < 0 \end{cases} \\ \left(x_0 - x_{e_{max}}\right)e^{(t-t_0)\left(A - Bh_{w,max}\right)} + x_{e_{max}}; & \begin{cases} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{cases} \\ \ldots; & \ldots \end{cases} \tag{C.44}$$

Next, consider the configurations that are not in (C.30) and $x_0 > 0$ and $x_f < 0$.

Assume that the system begins operating in mode A and then will switch to mode B. This implies that $p(t) < 0 \; \forall \; t \in \left[t_0, t_f\right]$ and $u(t) = 0$ until the switch of sign of $x(t)$ at time t_1. Under this assumption, the only x_{e_0}, $x_{e_{max}}$, and x_0 that drive the system from x_0 to 0 are described by (C.33) when $x_f = 0$. Therefore the only permissible configuration is the trivial case $x_0 = 0$.

Next, assume the initial operation is in mode B followed by a switch to mode A. This implies that $p(t) > 0 \; \forall \; t \in \left[t_0, t_f\right]$ and $u(t) = -x(t)h_{w,max}$ until the switch of sign of $x(t)$ at time t_1. Under this assumption, the configurations of x_{e_0}, $x_{e_{max}}$, and x_0 that drive the system from x_0 to 0 are (C.36) when $x_f = 0$. Although $x(t)$ approaches $x_{e_{max}}$ it does not reach it in finite time and therefore the only permissible configurations are those that satisfy (C.45).

$$x_{e_{max}} < 0 < x_0 \tag{C.45}$$

Because no configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f with $x_0 > 0$ and $x_f < 0$ begin in mode A, it is only necessary the switching behavior from mode B to mode A for $x_0 > 0$, $x_f < 0$, and $p(t) > 0$. The time t_1

that $x(t)$ changes sign is computed by (C.46) which is only finite and positive if (C.47) is satisfied which is guaranteed by $x_0 > 0$, $x_f < 0$, and (C.45).

$$\begin{aligned} x(t_1) &= \left(x_0 - x_{e_{max}}\right)e^{(t_1-t_0)(A-Bh_{w,max})} + x_{e_{max}} = 0 \\ \therefore t_1 &= \frac{1}{\left(A - Bh_{w,max}\right)}\ln\left(\frac{-x_{e_{max}}}{x_0 - x_{e_{max}}}\right) + t_0 \end{aligned} \tag{C.46}$$

$$0 < \left(\frac{-x_{e_{max}}}{x_0 - x_{e_{max}}}\right) < 1 \tag{C.47}$$

After t_1 the system switches, the dynamics are governed by mode A and reach the final goal state x_f at time t_f calculated by (C.48). The target state is only reached in finite time when condition (C.49) is satisfied by (C.45), $x_0 > 0$, and $x_f < 0$ and the new condition $x_f > x_{e_0}$. This yields condition (C.50) for the system to start in mode B and switch to mode A.

$$\begin{aligned} x\left(t_f\right) &= -x_{e_0}e^{(t_f-t_1)A} + x_{e\,0} = x_f \\ \therefore t_f &= \frac{1}{A}\ln\left(\frac{x_f - x_{e_0}}{-x_{e_0}}\right) + t_1 \end{aligned} \tag{C.48}$$

$$0 < \left(\frac{x_f - x_{e_0}}{-x_{e_0}}\right) < 1 \tag{C.49}$$

$$x_{e_0} < x_f < 0 < x_0 \tag{C.50}$$

In summary, since (C.33), (C.36), (C.40), (C.43), and (C.50) do not overlap, the operating mode can be determined *a priori*.

$$x(t) = \begin{cases} \left(x_0 - x_{e_0}\right)e^{(t-t_0)A} + x_{e_0}; & \left(t_0 \le t \le t_f\right) \wedge \begin{cases} 0 < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < 0 \end{cases} \\ \left(x_0 - x_{e_{max}}\right)e^{(t-t_0)(A-Bh_{w,max})} + x_{e_{max}}; & \left(t_0 \le t \le t_f\right) \wedge \begin{cases} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{cases} \\ \left(x_0 - x_{e_{max}}\right)e^{(t-t_0)(A-Bh_{w,max})} + x_{e_{max}}; & \left(t_0 \le t \le t_1\right) \wedge \left(x_{e_0} < x_f < 0 < x_0\right) \\ -x_{e_0}e^{(t-t_1)A} + x_{e_0}; & \left(t_1 < t_f\right) \wedge \left(x_{e_0} < x_f < 0 < x_0\right) \\ \dots; & \dots \end{cases} \tag{C.51}$$

Where

$$t_1 = \begin{cases} t_1 = \dfrac{1}{\left(A - Bh_{w,max}\right)} \ln\left(\dfrac{-x_{e_{max}}}{x_0 - x_{e_{max}}}\right) + t_0; & x_{e_0} < x_f < 0 < x_0 \\ \dots; & \dots \end{cases} \tag{C.52}$$

$$t_f = t_0 + \begin{cases} \dfrac{1}{A}\ln\left(\dfrac{x_f - x_{e_0}}{x_0 - x_{e_0}}\right); & \begin{cases} 0 < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < 0 \end{cases} \\ \dfrac{1}{\left(A - Bh_{w,max}\right)}\ln\left(\dfrac{x_f - x_{e_{max}}}{x_0 - x_{e_{max}}}\right); & \begin{cases} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{cases} \\ \dfrac{1}{A}\ln\left(\dfrac{x_f - x_{e_0}}{-x_{e_0}}\right) + t_1 - t_0; & x_{e_0} < x_f < 0 < x_0 \end{cases} \tag{C.53}$$

Finally, consider the configurations that are not in (C.30) and $x_0 < 0$ and $x_f > 0$.

Assume that the system begins operating in mode A and then will switch to mode B. This implies that $p(t) > 0 \; \forall \; t \in \left[t_0, t_f\right]$ and $u(t) = 0$ until the switch of sign of $x(t)$ at time t_1. Under this assumption, the only x_{e_0}, $x_{e_{max}}$, and x_0 that drive the system from x_0 to 0 are described by (C.40) when $x_f = 0$. Therefore the only permissible configuration is the trivial case $x_0 = 0$.

Next, assume the initial operation is in mode B followed by a switch to mode A. This implies that $p(t) < 0 \; \forall \; t \in \left[t_0, t_f\right]$ and $u(t) = -x(t)h_{w,max}$ until the switch of sign of $x(t)$ at time t_1. Under this assumption, the configurations of x_{e_0}, $x_{e_{max}}$, and x_0 that drive the system from x_0 to 0 are (C.43) when $x_f = 0$. Although $x(t)$ approaches $x_{e_{max}}$, it does not reach it in finite time and therefore the only permissible configurations are those that satisfy (C.54).

$$x_0 < 0 < x_{e_{max}} \tag{C.54}$$

Because no configurations of x_{e_0}, $x_{e_{max}}$, x_0, and x_f with $x_0 < 0$ and $x_f > 0$ begin in mode A, it is only necessary the switching behavior from mode B to mode A for $x_0 < 0$, $x_f > 0$, and $p(t) < 0$. The time t_1 that $x(t)$ changes sign is computed by (C.46) which is only finite and positive if (C.47) is satisfied which is guaranteed by $x_0 < 0$, $x_f > 0$, and (C.54).

After t_1 the system switches, the dynamics are governed by mode A and reach the final goal state x_f at time t_f calculated by (C.48). The target state is only reached in finite time when condition (C.49) is satisfied by (C.54), $x_0 < 0$, and $x_f > 0$ and the new condition $x_f < x_{e_0}$. This yields condition (C.55) for the system to start in mode B and switch to mode A.

$$x_0 < 0 < x_f < x_{e_0} \tag{C.55}$$

In summary, since (C.33), (C.36), (C.40), (C.43), (C.50), and (C.55) do not overlap, the operating mode can be determined *a priori*.

$$x(t) = \begin{cases} (x_0 - x_{e_0})e^{(t-t_0)A} + x_{e_0}; & (t_0 \le t \le t_f) \wedge \begin{cases} 0 < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < 0 \end{cases} \\ (x_0 - x_{e_{max}})e^{(t-t_0)(A-Bh_{w,max})} + x_{e_{max}}; & (t_0 \le t \le t_f) \wedge \begin{cases} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{cases} \\ (x_0 - x_{e_{max}})e^{(t-t_0)(A-Bh_{w,max})} + x_{e_{max}}; & (t_0 \le t \le t_1) \wedge \begin{cases} x_{e_0} < x_f < 0 < x_0 \\ x_0 < 0 < x_f < x_{e_0} \end{cases} \\ -x_{e_0}e^{(t-t_1)A} + x_{e_0}; & (t_1 < t \le t_f) \wedge \begin{cases} x_{e_0} < x_f < 0 < x_0 \\ x_0 < 0 < x_f < x_{e_0} \end{cases} \\ {}^{N}/_{A}; & o.w. \end{cases} \tag{C.56}$$

Where

$$t_1 = \frac{1}{(A - Bh_{w,max})} \ln\left(\frac{-x_{e_{max}}}{x_0 - x_{e_{max}}}\right) + t_0 \tag{C.57}$$

$$t_f = \begin{cases} \frac{1}{A}\ln\left(\frac{x_f - x_{e_0}}{x_0 - x_{e_0}}\right) + t_0\,; & \begin{cases} 0 < x_0 < x_f < x_{e_0} \\ x_{e_0} < x_f < x_0 < 0 \end{cases} \\ \frac{1}{(A - Bh_{w,max})}\ln\left(\frac{x_f - x_{e_{max}}}{x_0 - x_{e_{max}}}\right) + t_0; & \begin{cases} x_{e_{max}} < 0 < x_f < x_0 \\ 0 < x_{e_{max}} < x_f < x_0 \\ x_0 < x_f < x_{e_{max}} < 0 \\ x_0 < x_f < 0 < x_{e_{max}} \end{cases} \\ \frac{1}{A}\ln\left(\frac{x_f - x_{e_0}}{-x_{e_0}}\right) + t_1; & \begin{cases} x_{e_0} < x_f < 0 < x_0 \\ x_0 < 0 < x_f < x_{e_0} \end{cases} \\ {}^{N}/_{A}; & o.w. \end{cases} \tag{C.58}$$

In practice the control schedule describe above would not be strictly followed. If pumping water raises the block temperature to get closer to the set-point, then no water should be pumped in order to save energy. Additionally, this control algorithm is not guaranteed to be optimal, it has only been shown to meet the necessary, but not necessarily sufficient, conditions to be a locally optimal control. Once the final time has been reached, this control algorithm ceases to be appropriate and some other control algorithm must take over such as setting the flow to a level that results in the controlled equilibrium state equal to the desired final state. This post-objective control, described by (C.59) is only appropriate for a certain set of system parameters and final states such as those shown in (C.60).

$$f\left(x_f, u_{e_f}, t\right) = Ax_f + Bu_{e_f} + G = 0$$
$$\therefore u_{e_f} = \frac{-G - Ax_f}{B} \tag{C.59}$$

$$\left(\frac{-G - Ax_f}{B}\right)\left(\left(\frac{-G - Ax_f}{B}\right) - h_{w,max}x_f\right) \leq 0 \tag{C.60}$$

Bilinear formulation

Redefining the problem in the bilinear form of the system defined in the beginning of this appendix yields

$$f(x, u, t) = Ax(t) + Bx(t)u(t) + G; \quad x\left(t_f\right) = x_f \tag{C.61}$$

$$u(t) \in [0, u_{max}] \tag{C.62}$$

And,

$$A, B, b, G, x, u \in \mathbb{R}$$
$$A < 0; \quad B < 0; \quad 0 \leq u(t) \leq u_{max} \tag{C.63}$$

Since all the variables are scalar, scalar math can be used.

The partial derivatives and Hamiltonian then become

$$f_x = A + Bu(t); \quad f_u = Bx(t) \tag{C.64}$$

$$H(x, u, p; t) = \lambda_0 + p(t)(Ax(t) + Bx(t)u(t) + G) \tag{C.65}$$

The three necessary conditions for optimality from the minimum principle become

1) $\lambda_0 + p^* \cdot (Ax^* + Bx^*u^* + G) \leq \lambda_0 + p^* \cdot (Ax^* + Bx^*u + G)$
$\forall\ t \in \left[t_0, t_f\right]\ and\ \forall\ u \in [0, u_{max}]$
$p^*Bx^*u^* \leq p^*Bx^*u\ \forall\ t \in \left[t_0, t_f\right] and\ u \in [0, u_{max}]$

$p^{*}x^{*}u^{*} \geq p^{*}x^{*}u \; \forall \; t \in \left[t_0, t_f\right] and \; u \in [0, u_{max}]$

$$u^{*} = \begin{cases} 0, & \begin{cases} p^{*} > 0, & x^{*} < 0 \\ p^{*} < 0, & x^{*} > 0 \end{cases} \\ u_{max}, & \begin{cases} p^{*} > 0, & x^{*} > 0 \\ p^{*} < 0, & x^{*} < 0 \end{cases} \end{cases} \forall \; t \in \left[t_0, t_f\right] \tag{C.66}$$

2) $\lambda_0 + p(t)(Ax(t) + Bx(t)u(t) + G) = 0 \; \forall \; t \in \left[t_0, t_f\right]$

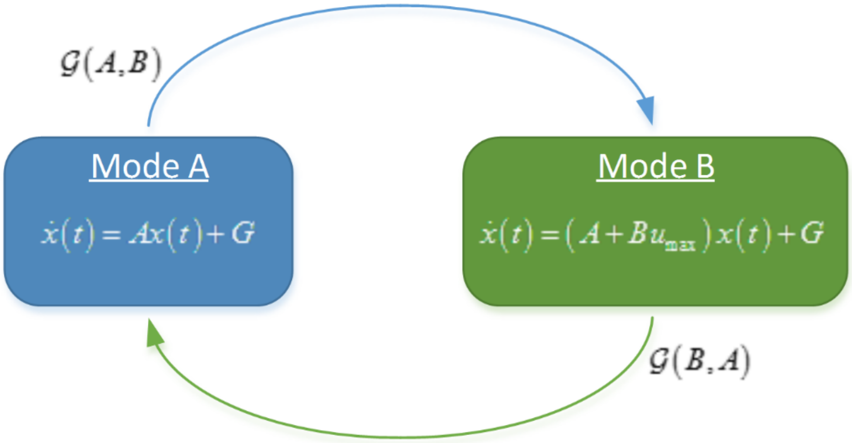
$$p(t) = \frac{-\lambda_0}{Ax(t) + Bx(t)u(t) + G} \tag{C.67}$$

3) $\dot{p}(t) = -p(t)\big(A + Bu(t)\big) \Rightarrow p(t) = c_1 e^{\int_{t_0}^{t} -(A+Bu(\tau))d\tau}$

$$p(t) = p(t_0)e^{\int_{t_0}^{t} -(A+Bu(\tau))d\tau} \tag{C.68}$$

∴ The sign of $p(t)$ does not change from $t_0 \leq t \leq t_f$ and is independent of $u(t)$ and $x(t)$ due to the facts $A < 0, B < 0$, and $u(t) \geq 0 \; \forall \; t \in \left[t_0, t_f\right]$.

Combining the state equation (C.61), the feedback function (C.66), and the knowledge that the sign of $p(t)$ does not change dynamically, it is possible to show that the system can be described by a hybrid automaton.



The dynamics of mode A described by (C.69) and reveal that the state will monotonically approach x_{e_0}.

$$x(t) = (x_0 - x_{e_0})e^{(t-t_0)A} + x_{e_0} \tag{C.69}$$

Likewise, the dynamics of mode B in (C.70) reveal that the state monotonically approaches $x_{e_{max}}$.

$$x(t) = (x_0 - x_{e_{max}})e^{(t-t_0)(A + B\, u_{max})} + x_{e_{max}} \tag{C.70}$$

By comparing the feedback equation (C.66) with the feedback equation (C.25) likewise for the state--co-state equations (C.67) and (C.26), the co-state dynamics (C.68) and (C.27), and the operating dynamics for mode A and B (C.69), (C.70), (C.28), and (C.29), it becomes apparent that the linear formulation and bilinear formulation of the minimum time problem result in exactly the same mathematical program where the only symbol difference is $-h_{w,max} = u_{max}$.

REFERENCES

AASHTO. (2008). *Bridging the Gap: Restoring and Rebuilding the Nation's Bridges*. Washington, DC, 68.

Abraham, S., and Efford, R. J. (2004). *Final Report on the August 14, 2003 Blackout in the United States and Canada - Causes and Recommendations*. 238.

Alzraiee, H., Moselhi, O., and Zayed, T. (2012). "A Hybrid Framework for Modeling Construction Operations Using Discrete Event Simulation and System Dynamics." *Construction Research Congress 2012*, 1063–1073.

Architecture Technology Corporation. (2003). *WIRL: Wireless Integrated Routing Link*. Eden Prairie, MN.

Aswani, A., Master, N., Taneja, J., Culler, D., and Tomlin, C. (2012). "Reducing Transient and Steady State Electricity Consumption in HVAC using Learning-Based Model Predictive Control." *Proceedings of the IEEE*, 100(1), 240–253.

Bachman, R. J., Woolaver, D. A., and Powell, M. D. (2007). "Sea Fighter (FSF-1) Seakeeping Measurements." Naval Surface Warefare Center Carderock Division, West Bethesda, MD.

Banerjee, B. A., Venkatasubramanian, K. K., Mukherjee, T., and Gupta, S. K. S. (2012). "Ensuring Safety , Security , and Cyber – Physical Systems." *Proceedings of the IEEE*, 100(1), 283–299.

Beeby, S., and White, N. (2010). *Energy harvesting for autonomous systems*. Artech House, Norwood, Mass., ix, 292 p.

Belinfante, J. G., and Kolman, B. (1989). *A Survey of Lie Groups and Lie Algebras with Applications and Computational Methods*. Society for Industrial and Applied Mathematics, 159.

Belkin International Inc. (2013). "Home Automation | Belkin USA Site." <http://www.belkin.com/us/Products/c/home-automation/> (Oct. 16, 2013).

Bemporad, A. (2013). "Hybrid Toolbox for MATLAB."

Bemporad, A., Morari, M., Garulli, A., and Tesi, A. (1999). *Robustness in identification and control*. (A. Garulli and A. Tesi, eds.), Springer London, London, 207–226.

Benedetto, M. Di. (2008). “Automatic verification of wireless control in a mining ventilation system.” *… , 2008. CASE 2008. …*, Ieee, 858–863.

Bennett, R., Hayes-Gill, B., Crowe, J. A., Armitage, R., Rodgers, D., and Hendroff, A. (1999). “Wireless monitoring of highways.” *Smart Systems for Bridges, Structures, and Highways*, Newport Beach, CA, 173–182.

Bergman, T. L. (2011). *Fundamentals of heat and mass transfer*. John Wiley, Hoboken, NJ, 1048.

Branicky, M. (1998). “A unified framework for hybrid control: Model and optimal control theory.” *Automatic Control, IEEE …*, 43(1), 31–45.

Brezinski, C. (2002). *Computational Aspects of Linear Control*. Dordrecht: Kluwer Academic Publishers.

Bridge Diagnostics Inc. (2012). *STS WiFi Wireless Structural Testing System*. Boulder, CO.

Bryson, A. E., and Ho, Y.-C. (1975). *Applied optimal control: optimization, estimation, and control*. Hemisphere Pub. Corp. ; distributed by Halsted Press, Washington : New York, 481 p.

Burnham, K. J., James, D. J. G., and Goodhart, S. G. (1994). “Bilinear self-tuning control of a high temperature heat treatment plant.” *IEE Proceedings - Control Theory and Applications*, 141(1), 12–18.

Calhoun, B., and Lach, J. (2012). “Body sensor networks: A holistic approach from silicon to users.” *Proceedings of the IEEE*, 100(1), 91–106.

Camacho, E. F., and Bordons, C. (1999). *Model predictive control*. Springer, 280.

Camponogara, E., Jia, D., Krogh, B. H., and Talukdar, S. (2002). “Distributed model predictive control.” *Control Systems, IEEE*, 22(1), 44–52.

Chen, D., Nixon, M., Mok, A., and Service), S. (Online. (2010). *WirelessHART™ Real-Time Mesh Network for Industrial Automation*. Springer Science+Business Media, LLC, Boston, MA.

Chipka, J., Lisicki, A., and Nguyen, C. (2013). “Experimental Characterization and Predictive Modeling of a Residential-Scale Wind Turbine.” *Conference Proceedings of the Society for Experimental Mechanics Series 2013*, Springer New York, 521–533.

Christopoulos, C., Filiatrault, A., and Bertero, V. V. (2006). *Principles of passive supplemental damping and seismic isolation*. IUSS Press, 480.

Chu, S. (2009). "Secretary Chu's Remarks at the Cincinnati State Technical and Community College Commencement." *Secretary*, U.S. Department of Energy, Cincinnati.

Cisco Systems Inc. (2007). *20 Myths of Wi-Fi Interference: Dispel Myths to Gain High-Performing and Reliable Wireless*. San Jose, CA, 9.

Coats, J. (1747). *A New Dictionary of Heraldry*. Ptd. for Aaron Ward, London, 376.

"Code Composer Studio." (2013). Texas Instruments Incorporated.

Coffman, E. G. J., Elphick, M. J., and Shoshani, A. (1971). "System Deadlocks." *Computing Surveys*, 3(2), 67–78.

Cortés, C. E., Sáez, D., Milla, F., Núñez, A., and Riquelme, M. (2010). "Hybrid predictive control for real-time optimization of public transport systems' operations based on evolutionary multi-objective optimization." *Transportation Research Part C: Emerging Technologies*, Elsevier Ltd, 18(5), 757–769.

Crossbow Technology. (2007). "Imote2 Datasheet, Document Part Number: 6020-0117-02 Rev A."

Deng, K., Sun, Y., Chakraborty, A., Lu, Y., Brouwer, J., and Mehta, P. G. (2013). "Optimal scheduling of chiller plant with thermal energy storage using mixed integer linear programming." *American Control …*, IEEE, Washington, D.C., 2958–2963.

Derler, P., Lee, E. a., and Vincentelli, a. S. (2012). "Modeling Cyber–Physical Systems." *Proceedings of the IEEE*, 100(1), 13–28.

Dollhopf, R., and Durno, M. (2011). "Kalamazoo River\Enbridge Pipeline Spill 2010." *International Oil Spill Conference Proceedings*, Portland, OR, 1–7.

Dunnington, L., Stevens, H., and Grater, G. (2003). *Integrated Engineering Plant for Future Naval Combatants-Technology Assessment and Demonstration Roadmap*. *(S3) Technology Symposium Change, Challenges & Constants*, Fairfax, VA, MSD–50–TR–2003/01.

Ekman, M. (2005). "Modeling and control of bilinear systems: application to the activated sludge process." Uppsala University.

Elbeheiry, E. M. (2001). "Suboptimal Bilinear Control Methods Applied to Suppressing Car Vibrations." *Journal of Vibration and Control*, 7(2), 279–306.

Elbeheiry, E. M., Karnopp, D. C., Elaraby, M. E., and Abdelraaouf, A. M. (1995). “Advanced Ground Vehicle Suspension Systems - A Classified Bibliography.” *Vehicle System Dynamics*, 24(3), 231–258.

Electric Imp Inc. (2013). “electric imp.” <http://electricimp.com/> (Oct. 16, 2013).

Elhaddad, W. M., and Johnson, E. A. (2013). “Hybrid MPC: An Application to Semiactive Control of Structures.” *Conference Proceedings of the Society for Experimental Mechanics*, F. N. Catbas, S. Pakzad, V. Racic, A. Pavic, and P. Reynolds, eds., Springer New York, New York, NY, 27–36.

Elliott, D. L. (2007). “Bilinear Systems.” *Wiley Encyclopedia of Electrical and Electronics Engineering*, John Wiley & Sons, Inc.

Elliott, D. L. (2009). *Bilinear control systems: matrices in action*. (S. S. Antman, L. Sirovich, and J. E. Marsden, eds.), Springer, Dordrecht [Netherlands] ; New York, ix, 280 p.

Estrin, D., Heidemann, J., Kumar, S., and Rey, M. (1999). “Next Century Challenges: Scalable Coordination in Sensor Networks.” *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking*, ACM, Seattle, WA, 263–270.

Etherios Inc. (2013). “Device Cloud: Driving the Internet of ANYthing.” <http://www.etherios.com/products/devicecloud/> (Oct. 16, 2013).

Fawal, H. E., Georges, D., and Bornard, G. (n.d.). “Optimal control of complex irrigation systems via decomposition-coordination and the use of augmented Lagrangian.” *SMC’98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.98CH36218)*, IEEE, 3874–3879.

Fink, J., Ribeiro, a., and Kumar, V. (2012). “Robust Control for Mobility and Wireless Communication in Cyber–Physical Systems With Application to Robot Teams.” *Proceedings of the IEEE*, 100(1), 164–178.

Floudas, C. a., and Ciric, a. R. (1989). “Strategies for overcoming uncertainties in heat exchanger network synthesis.” *Computers & Chemical Engineering*, 13(10), 1133–1152.

Fok, C., Roman, G., and Lu, C. (2005). “Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications.” *Distributed Computing Systems, 2005. ICDCS 2005. Proceedings. 25th IEEE International Conference on*, IEEE, 653–622.

Galassi, M., and Theiler, J. (2013). *GNU Scientific Library Reference Manual*.

Giorgetti, N., Bemporad, a., Tseng, H. E., and Hrovat, D. (2006). “Hybrid model predictive control application towards optimal semi-active suspension.” *International Journal of Control*, 79(5), 521–533.

Goebel, R., Sanfelice, R., and Teel, A. (2009). “Hybrid dynamical systems.” *Control Systems, IEEE*, IEEE, (April), 28 –93.

Graf, H. B., Hermanns, H., Kulshrestha, J., Peter, J., Vahldiek, A., and Vasudevan, A. (2011). “A Verified Wireless Safety Critical Hard Real-Time Design.” *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, IEEE, Lucca, Italy, 1–9.

Graichen, K., and Käpernick, B. (2011). “A Real-Time Gradient Method for Nonlinear Model Predictive Control.” *Frontiers of Model Predictive Control*, T. Zheng, ed., InTech, 9–28.

Hamano, J., Torvalds, L., and et al. (2013). “Git.”

Han, S. (1977). “A globally convergent method for nonlinear programming.” *Journal of optimization theory and applications*, 22(3), 297–309.

HART Communication Foundation. (2011). “WirelessHART Achieves Widespread Global Acceptance [Press release].”

Holmes, J. D. (2007). *Wind Loading of Structures*. Taylor and Francis, New York, NY, 392.

Horowitz, P., and Hill, W. (1989). *The art of electronics*. Cambridge University Press, Cambridge [England] ; New York, xxiii, 1125 p.

Housner, G. W., Bergman, L. A., Caughey, T. K., Chassiakos, A. G., Claus, R. O., Masri, S. F., Skelton, R. E., Soong, T. T., Spencer, B. F., and Yao, J. T. P. (1997). “Structural Control: Past, Present, and Future.” *Journal of Engineering Mechanics*, ASCE, 123(9), 897–971.

IEEE Computer Society. (2006). *IEEE Standard for Information technology-Telecommunications and information exchange between systems- Local and metropolitan area networks- Specific requirements--Part 15.4: Wireless MAC and PHY Specifications for Low-Rate WPANs*. New York, NY, 323.

IFTTT Inc. (2013). “IFTTT / Put the internet to work for you.” <https://ifttt.com/> (Oct. 16, 2013).

Ihler, E., Zaglauer, H., and Herold-Schmidt, U. (2000). "Integrated wireless piezoelectric sensors." *Proc. SPIE 3991, Smart Structures and Materials 2000: Industrial and Commercial Applications of Smart Structures Technologies, 44*, 44–51.

Inaudi, D., and Manetti, L. (2009). "Reinforced Concrete Corrosion Wireless Monitoring System." *4th International Conference on Structural Health Monitoring on Intelligent Infrastructure (SHMII-4) 2009*, Zurich, 1–10.

Jo, H., Sim, S., Nagayama, T., and Spencer, J. B. F. (2012). "Development and Application of High-Sensitivity Wireless Smart Sensors for Decentralized Stochastic Modal Identification." *Journal of Engineering …*, 138(6), 683–694.

Johnson, M., Healy, M., van de Ven, P., Hayes, M. J., Nelson, J., Newe, T., and Lewis, E. (2009). "A comparative review of wireless sensor network mote technologies." *Sensors, 2009 IEEE*, IEEE, 1439–1442.

Kane, M. B., and Lynch, J. P. (2012). "An Agent-Based Model-Predictive Controller for Chilled Water Plants using Wireless Sensor and Actuator Networks." *American Control Conference (ACC), 2012*, IEEE, Montreal, Canada, 1192–1198.

Kane, M. B., Lynch, J. P., and Law, K. (2011). "Market-based control of shear structures utilizing magnetorheological dampers." *American Control Conference (ACC), 2011*, 2498–2503.

Kane, M. B., Lynch, J. P., and Zimmerman, A. T. (2011). "Decentralized agent-based control of chilled water plants using wireless sensor and actuator networks." *Resilient Control Systems (ISRCS), 2011 4th International Symposium on*, 131–136.

Kelman, A., and Borrelli, F. (2011). "Bilinear Model Predictive Control of a HVAC System Using Sequential Quadratic Programming." *18th IFAC World Congress*, International Federation of Automatic Control (IFAC), Milano, Italy, 9869–9874.

Kempton, W., and Tomić, J. (2005). "Vehicle-to-grid power implementation: From stabilizing the grid to supporting large-scale renewable energy." *Journal of Power Sources*, 144(1), 280–294.

Kim, B. J., Alleman, J. E., Gee, C. S., and Bandy, J. T. (1991). *Use of Programmable Logic Controllers To Automate Control and Monitoring of U.S. Army Wastewater Treatment Systems*. *Security*, Champaign, 98.

Kim, J., and Lynch, J. P. (2012). "Autonomous Decentralized System Identification by Markov Parameter Estimation Using Distributed Smart Wireless Sensor Networks." *J. Eng. Mech.*, 138(5), 478–490.

Kim, J., and Nadukuru, S. (2012). "Assessment of the Behavior of Buried Concrete Pipelines Subjected to Ground Rupture: Experimental Study." *Journal of Pipeline …*, (February), 8–16.

Kirk, D. (2004). *Optimal control theory: an introduction*. Dover Publications, Mineola, N.Y, 452.

Kottapalli, V. A., Kiremidjian, A. S., Lynch, J. P., Carryer, E., Kenny, T. W., Law, K. H., and Lei, Y. (2003). "Two-tiered Wireless Sensor Network Architecture for Structural Health Monitoring." *Proc. SPIE 5057, Smart Structures and Materials 2003: Smart Systems and Nondestructive Evaluation for Civil Infrastructures*, Citeseer.

Kurata, M., Kim, J., Lynch, J. P., Linden, G. W. Van Der, Sedarat, H., Thometz, E., Hipley, P., and Sheng, L. -H. (2012). "Internet-Enabled Wireless Structural Monitoring Systems: Development and Permanent Deployment at the New Carquinez Suspension Bridge." *Journal of Structural Engineering*, 139(10), 1668–1702.

Lee, S. H., Kong, J., and Seo, J. H. (1997). "Observers for bilinear systems with unknown inputs and application to superheater temperature control." *Control Engineering Practice*, 5(4), 493–506.

Linderman, L. E. (2013). "Smart Wireless Control of Civil Structures." University of Illinois at Urbana-Champaign.

Liu, J., de la Pena, D. M., and Christofides, P. D. (2009). "Distributed model predictive control of nonlinear systems with input constraints." *American Control Conference, 2009. ACC '09.*, 2319–2326.

Livadas, C., Lygeros, J., and Lynch, N. a. (2000). "High-level modeling and analysis of the traffic alert and collision avoidance system (TCAS)." *Proceedings of the IEEE*, 88(7), 926–948.

Logan, K. P., Cartes, D. a., and Srivastava, S. K. (2007). "Fault Detection, Diagnostics, and Prognostics: Software Agent Solutions." *IEEE Transactions on Vehicular Technology*, 56(4), 1613–1622.

Loh, C. H., Lynch, J. P., Lu, K. C., Wang, Y., Chang, C. M., Lin, P. Y., and Yeh, T. H. (2007). "Experimental verification of a wireless sensing and control system for structural control using MR dampers." *Earthquake Engineering and Structural Dynamics*, Citeseer, 36(10), 1303–1328.

Los Alamos National Security, L. (2013). “The Weather Machine: Los Alamos National Laboratory.” <http://environweb.lanl.gov/weathermachine/> (Oct. 16, 2013).

Lunze, J., and Lamnabhi-Lagarrigue, F. (2009). “Handbook of hybrid systems control: theory, tools, applications.” Cambridge University Press, Cambridge, UK ; New York.

Lynch, J. P., and Law, K. H. (2002). “Market-based control of linear structural systems.” *Earthquake Engineering & Structural Dynamics*, 31(10), 1855–1877.

Lynch, J. P., and Law, K. H. (2004). “Decentralized energy market-based structural control.” *Structural Engineering and Mechanics*, 17(3-4).

Lynch, J. P., Law, K. H., Kiremidjian, A. S., Kenny, T. W., Carryer, E., and Partridge, A. (2001). “The design of a wireless sensing unit for structural health monitoring.” *Proceedings of the 3rd International Workshop on Structural Health Monitoring*, Stanford, CA.

Lynch, J. P., and Loh, K. J. (2006). “A summary review of wireless sensors and sensor networks for structural health monitoring.” *The Shock and vibration digest*, 38(2), 91–128.

Lynch, J. P., Sundararajan, A., Law, K. H., Kiremidjian, A. S., and Carryer, E. (2003). “Power-efficient data management for a wireless structural monitoring system.” *4th International Workshop on Structural Health Monitoring*, Stanford, California, 1177–1184.

Lynch, J. P., Sundararajan, A., Law, K. H., Kiremidjian, A. S., and Carryer, E. (2004). “Embedding damage detection algorithms in a wireless sensing unit for attainment of operational power efficiency.” *Smart Materials and Structures, IOP*, 13(4), 800–810.

Lynch, J. P., Wang, Y., Swartz, R. A., Lu, K.-C., and Loh, C.-H. (2008). “Implementation of a closed-loop structural control system using wireless sensor networks.” *Journal of Structural Control and Health Monitoring*.

Maestre, J. M., Raso, L., Overloop, P. J. Van, and Schutter, B. De. (2012). “Distributed Tree-Based Model Predictive Control on an Open Water System.” *American Control Conference, 2012*, IEEE, Montreal, Canada.

Maróti, M., Kusy, B., Simon, G., and Lédeczi, Á. (2004). “The flooding time synchronization protocol.” *Proceedings of the 2nd International*

Conference on Embedded Networked Sensor Systems, ACM Press, Baltimore, MD, 39–49.

Mascarenas, D. L., Flynn, E. B., Todd, M. D., Overly, T. G., Farinholt, K. M., Park, G., and Farrar, C. R. (2010). "Experimental studies of using wireless energy transmission for powering embedded sensor nodes." *Journal of Sound and Vibration*, Elsevier, 329(12), 2421–2433.

Maturana, F. P., Staron, R. J., and Hall, K. H. (2005). "Methodologies and tools for intelligent agents in distributed control." *Intelligent Systems, IEEE*, 20(1), 42–49.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M. (2000). "Constrained model predictive control: Stability and optimality." *Automatica*, PERGAMON PRESS, 36(6), 789–814.

Mccullagh, J., Peterson, R. L., Galchev, T., Gordenker, R., Zhang, Y., Lynch, J., and Najafi, K. (2012). "SHORT-TERM AND LONG-TERM TESTING OF A VIBRATION HARVESTING SYSTEM FOR BRIDGE HEALTH MONITORING." *PowerMEMS*, Atlanta, 5–8.

Microstrain Inc. (2012). "Wireless System Networks | Microstrain." <http://www.microstrain.com/wireless/systems>.

Mitchell, K., Dang, N., Liu, P., Rao, V. S., and Pottinger, H. J. (2001). "Web-controlled wireless network sensors for structural health monitoring." *Smart Structures and Materials – Smart Electronics and MEMS*, Newport Beach, CA, 234–243.

Mohler, R. (1970). "Natural bilinear control processes." *… Science and Cybernetics, IEEE Transactions on*, 6(3), 192–197.

Morari, M., and Lee, J. H. (1999). "Model predictive control: past, present and future." *Computers & Chemical Engineering*, 23(4-5), 667–682.

Nagayama, T., Spencer Jr, B., Mechitov, K., and Agha, G. (2009). "Middleware services for structural health monitoring using smart sensors." *Smart Structures and Systems*, 5(2), 119–137.

Naidu, D., and Rieger, C. (2011). "Advanced control strategies for heating, ventilation, air-conditioning, and refrigeration systems—An overview: Part I: Hard control." *HVAC&R Research*, 17(1), 2–21.

National Instruments Inc. (2012). "Wireless Measurement Device Selection Guide."

<http://sine.ni.com/np/app/main/p/ap/global/lang/en/pg/1/sn/n24: Wireless/fmid/2988/>.

Negenborn, R. R., De Schutter, B., and Hellendoorn, H. (2006). "Multi-Agent Model Predictive Control of Transportation Networks." *2006 IEEE International Conference on Networking, Sensing and Control*, Ieee, 296–301.

Nocedal, J., Wright, S. J., and Robinson, S. M. (1999). *Numerical Optimization*. (P. Glynn and S. Robinson, eds.), Springer-Verlag, New York, 634.

NTSB. (2007). *Collapse of I-35W Highway Bridge Minneapolis, Minnesota August 1, 2007*. 178.

Ogonowski, Z. (2011). "Two-state bilinear predictive control for hot-water storage tank." *Proceedings of the 18th International Conference on Process Control*, M. Fikar and M. Kvasnica, eds., Tatranska Lomnica, Slovakia, 408–414.

Oldewurtel, F., Parisio, A., Jones, C., Morari, M., Gyalistras, D., Gwerder, M., Stauch, V., Lehmann, B., and Wirth, K. (2010). "Energy efficient building climate control using stochastic model predictive control and weather predictions." *American Control Conference, 2010*, IEEE.

Omega Engineering Inc. (2012). "Wireless Sensors, Transmitters, Receivers, Meters & Controllers." <http://www.omega.com/toc_asp/subsectionsc.asp?book=DAS&subsection=K01>.

Overly, T. G. S., Park, G., Farinholt, K. M., and Farrar, C. R. (2008). "Development of an Extremely Compact Impedance-Based Wireless Sensing Device." *Smart Materials and Structures*, 17(6), 065011.

Özdemir, T., Goykhman, Y., Oberdier, L., and Lynch, J. (2010). "Smart antenna technology for structural health monitoring applications." *Proc. SPIE 7649, Nondestructive Characterization for Composite Materials, Aerospace Engineering, Civil Infrastructure, and Homeland Security*, P. J. Shull, A. A. Diaz, and H. F. Wu, eds., SPIE--The International Society for Optical Engineering, San Diego, CA, 76490M–76490M–7.

Pajic, M., and Mangharam, R. (2012). "Cyber–Physical Modeling of Implantable Cardiac Medical Devices." *Proceedings of the IEEE*, 100(1), 122–137.

Peters, B. (2005). "Sensing without wires." *Machine Design*, Cleveland, OH.

Pinoccio Inc. (2013). “Projects., Pinoccio - Wireless microcontroller for web-enabled DIY.” <http://pinocc.io/> (Oct. 16, 2013).

Piñón, S., Camacho, E. F., Kuchen, B., and Peña, M. (2005). “Constrained predictive control of a greenhouse.” *Computers and Electronics in Agriculture*, 49(3), 317–329.

Pita, E. G. (2001). *Air conditioning principles and systems*. Prentice Hall, 524.

Ploplys, N. J., Kawka, P. A., and Alleyne, A. G. (2004). “Closed-loop control over wireless networks.” *IEEE Control Systems Magazine*, 24(3), 58–71.

Qin, Y., Qiu, N., and Jia, L. (2009). “Research for Spatio-Temporal Modeling Approach of Railway Safety System Based on Hybrid Cellular Automata.” *Logistics*, American Society of Civil Engineers, 4432–4439.

Quickfilter Technologies. (2009). “QF4A512 Datasheet.” Quickfilter Technologies Inc., Allen, Texas.

Del Re, L., Chapuis, J., and Nevistik, V. (1993). “Predictive control with embedded feedback linearization for bilinear plants with input constraints.” *Decision and Control, 1993., Proceedings of the 32nd IEEE Conference on*, San Antonio, TX, 2984 – 2989 vol.4.

Redfern, A., Koplow, M., and Wright, P. (2006). “Design architecture for multi-zone HVAC control systems from existing single-zone systems using wireless sensor networks.” *Proc. SPIE 6414, Smart Structures, Devices, and Systems III*, S. F. Al-Sarawi, ed., Adelaide, Australia, 64140Y–64140Y–8.

Reeves, G. E. (1998). *What Really Happened on Mars?* Pasadena, Calif.

Rice, J. A., and Spencer Jr., B. F. (2008). “Structural health monitoring sensor development for the Imote2 platform.” *Proceedings of the SPIE Smart Structures/NDE 2008*, San Diego, CA.

Rohani, M., and Hadi Afshar, M. (2013). “Sewer Networks Optimization Using Cellular Automata.” *Studies in Engineering and Technology*, 1(1).

Rossman, L. A. (2008). “EPANET 2.” U.S. Environmental Protection Agency (EPA).

Schaller, R. R. (1997). “Moore’s law: past, present and future.” *Spectrum, IEEE*, 34(6), 52–59.

Scherer, C., and Weiland, S. (2000). *Linear matrix inequalities in control. … Dutch Institute for Systems and Control …*, Delft, The Netherlands, 60.

Scholze, R. J., and Zaghloul, H. H. (2001). *Assessment of SCADA Technology Applications to Automate U.S. Army Water and Wastewater Sanitary Systems*. *Evaluation*, CHAMPAIGN, 28.

Scruggs, J. T., Taflanidis, A. A., and Iwan, W. D. (2007). "Non-linear stochastic controllers for semiactive and regenerative systems with guaranteed quadratic performance bounds—Part 1: State feedback control." *Structural Control and …*, 14(April 2006), 1101–1120.

Seman III, A. J., Toomey, K., and Lang, S. (2003). "Reduce's Ship's Crew-By Virtual Presence (RSVP) Advanced Technology Demonstration (ATD)." *Final technical rept. 1 Sep 1998-1 Oct 2001*, Naval Surface Warfare Center Carderock Division, Bethesda, MD.

Seth, S., Lynch, J. P., and Tilbury, D. M. (2005). "Wirelessly networked distributed controllers for real-time control of civil structures." *American Control Conference, 2005. Proceedings of the 2005*, 2946–2952 vol. 4.

Sha, L., Gopalakrishnan, S., Liu, X., and Wang, Q. (2008). "Cyber-Physical Systems: A New Frontier." *2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (sutc 2008)*, Ieee, 1–9.

Shannon, C. E. (1928). "A mathematical theory of communication." *Bell Systems Technical Journal*, 27, 379–423, 623–656.

Siemens AG. (2012). "WirelessHART - Innovation for the process industry." <http://www.automation.siemens.com/w1/automation-technology-wirelesshart-18957.htm>.

Slaughter, S. B., Cheung, M. C., Sucharski, D., and Cowper, B. (1997). "State of the Art in Hull Monitoring Systems." *Report No. SSC-401*, Ship Structure Committee, Washington, DC.

Sohraby, D. K., Minoli, D., and Znati, T. (2007). *Wireless sensor networks*. John Wiley & Sons, Inc., Hoboken, NJ, 307.

Southwest Windpower Inc. (2000). *Whisper 500 Owners Manual*. Flagstaff, AZ, 46.

Spencer, B. F. (1997). "Controlling buildings: a new frontier in feedback." *IEEE Control Systems Magazine*, Institute of Electrical and Electronics Engineers, Inc, 445 Hoes Ln, Piscataway, NJ, 08854-1331, USA, 17(6), 19–35.

Spencer, B. F. J., and Nagarajaiah, S. (2003). "State of the Art of Structural Control." *Journal of structural engineering*, ASCE, 129(7), 845–856.

Spencer, B. F., Ruiz-Sandoval, M. E., and Kurata, N. (2004). “Smart sensing technology: opportunities and challenges.” *Journal of Structural Control and Health Monitoring*, 11(4), 349–368.

Sridhar, S., Hahn, A., and Govindarasu, M. (2012). “Cyber–Physical System Security for the Electric Power Grid.” *Proceedings of the IEEE*, 100(1), 210–224.

Srivastava, S. K., Cartes, D. A., Maturana, F., Ferrese, F., Pekala, M., Zink, M., Meeker, R., Carnahan, D., Staron, R., Scheidt, D., and Huang, K. (2008). “A Control System Test Bed for Demonstration of Distributed Computational Intelligence Applied to Reconfi guring Heterogeneous Systems.” *IEEE Instrumentation & Measurement Magazine*, 11(1), 30–37.

Straser, E. G., Kiremidjian, A. S., Meng, T. H., and Redlefsen, L. (1998). “A modular, wireless network platform for monitoring structures.” *16th Interational Modal Analysis Conference (IMAC)*, SEM, Santa Barbara, CA, 450–456.

Strickland, E. (2011). “24 Hours at Fukushima.” *IEEE Spectrum*.

Strzelecki, R., and Benysek, G. (2008). *Power electronics in smart electrical energy networks*. Springer, 414.

Susumpow, T., and Fujino, Y. (1995). “Active control of multimodal cable vibrations by axial support motion.” *Journal of engineering mechanics*, 121(9), 964–972.

Swartz, R. A., Jung, D., Lynch, J. P., Wang, Y., Shi, D., and Flynn, M. P. (2005). “Design of a wireless sensor for scalable distributed in-network computation in a structural health monitoring system.” *5th International Workshop on Structural Health Monitoring*, Stanford, CA.

Swartz, R. A., and Lynch, J. P. (2007). “Partial Decentralized Wireless Control Through Distributed Computing for Seismically Excited Civil Structures: Theory and Validation.” *American Control Conference, 2007. ACC ’07*, 2684–2689.

Swartz, R. A., Wang, Y., Lynch, J. P., Law, K. H., and Loh, C. (2010). “EXPERIMENTAL VALIDATION OF MARKET-BASED CONTROL USING WIRELESS SENSOR AND ACTUATOR NETWORKS.” *Relation*, Citeseer, 10(1.59), 5825.

Swartz, R., and Lynch, J. P. (2009). “Strategic Network Utilization in a Wireless Structural Control System for Seismically Excited Structures.” *Journal of structural engineering*, ASCE, 135(5), 597–608.

Swartz, R., Lynch, J., Zerbst, S., Sweetman, B., and Rolfes, R. (2010). "Structural Monitoring of Wind Turbines Using Wireless Sensor Networks." *Smart Structures and Systems*, 6(3), 1–8.

Swartz, R., Zimmerman, A. T., Lynch, J. P., Rosario, J., Brady, T., Salvino, L., and Law, K. H. (2010). "Hybrid wireless hull monitoring system for naval combat vessels." *Structure and Infrastructure Engineering*, Taylor & Francis, 8(7), 621–638.

Sztipanovits, J., Koutsoukos, X., Karsai, G., Kottenstette, N., Antsaklis, P., Gupta, V., Goodwine, B., and Baras, J. (2012). "Toward a Science of Cyber–Physical System Integration." *Proceedings of the IEEE*, 100(1), 29–44.

Taylor, S. G., Farinholt, K. M., Park, G., Farrar, C. R., Flynn, E. B., Mascarenas, D. L., and Todd, M. D. (2009). "Wireless impedance device for electromechanical impedance sensing and low-frequency vibration data acquisition." *Proc. SPIE 7292, Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2009*, (D. K. Lindner, Y. Bar-Cohen, M. Ahmadian, Z. Ounaies, B. K. Henderson, M. Tomizuka, H. F. Wu, V. K. Varadan, N. G. Meyendorf, T. Kundu, K. J. Peters, A. A. Diaz, M. B. McMickell, J. Li, M. N. Ghasemi-Nejhad, T. Wallmersperger, P. J. Shull, W. Ecke, and D. W. Vogel, eds.), 7292, 729228–729228–12.

Texas Instruments. (2009). "Application Note AN060 Security on TI IEEE 802.15.4 Compliant RF Devices." *Security*, Dallas, Texas, United States.

The National Academy of Engineering. (2012). "Restore and improve urban infrastructure - Engineering Challenges." *The National Academy of Engineering of the National Academies - Grand Challenges for Engineering*, <http://www.engineeringchallenges.org/cms/8996/9136.aspx> (Jan. 1, 2014).

ThingWorx. (2013). "ThingWorx - Internet of Things and M2M Application Platform." <http://www.thingworx.com/> (Oct. 16, 2013).

TinyOS Alliance. (2012). "TinyOS Home Page." <http://www.tinyos.net/>.

U.S. Department of Energy. (2011). *2010 Buildings Energy Data Book*. *Energy*, (D&R International Ltd., ed.), Pacific Northwest National Laboratory, Silver Spring, MD.

Vieira, M. A. M., Coelho, C.N., J., da Silva, D.C., J., and da Mata, J. M. (2003). "Survey on wireless sensor network devices." *Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA '03. IEEE Conference*, IEEE, 537–544.

Wagner, B. (2007). “All-Electric Ship Could Begin to Take Shape By 2012.” *National Defense*, National Defense Industrial Association.

Wang, Y. (2009). “Time-delayed dynamic output feedback H_infin controller design for civil structures: A decentralized approach through homotopic transformation.” *Structural Control and Health Monitoring*, 9999(9999), n/a.

Wang, Y., and Boyd, S. (2010). “Fast Model Predictive Control Using Online Optimization.” *Control Systems Technology, IEEE Transactions on*, 18(2), 267–278.

Wang, Y., and Law, K. H. (2011). “Structural Control with Multi-Subnet Wireless Sensing Feedback: Experimental Validation of Time-Delayed Decentralized H_∞ Control Design.” *Advances in Structural Engineering*, 14(1), 25–39.

Wang, Y., Lynch, J. P., and Law, K. H. (2009). “Decentralized Hinf; controller design for large-scale civil structures.” *Earthquake Engineering & Structural Dynamics*, John Wiley & Sons, 38(3), 377–401.

Wang, Y., Swartz, A., Lynch, J. P., Law, K. H., Lu, K.-C., and Loh, C.-H. (2006). “Wireless feedback structural control with embedded computing.” *Proceedings of SPIE--11th International Symposium on Nondestructive Evaluation for Health Monitoring and Diagnostics*, San Diego, CA.

Wang, Y., Swartz, R. A., Lynch, J. P., Law, K. H., Lu, K.-C., and Loh, C.-H. (2007). “Decentralized civil structural control using real-time wireless sensing and embedded computing.” *Smart Structures and Systems*, 3(3), 321–340.

Witrant, E., D’Innocenzo, a., Sandou, G., Santucci, F., Di Benedetto, M. D., Isaksson, a. J., Johansson, K. H., Niculescu, S.-I., Olaru, S., Serra, E., Tennina, S., and Tiberi, U. (2010). “Wireless ventilation control for large-scale systems: The mining industrial case.” *International Journal of Robust and Nonlinear Control*, 20(2), 226–251.

Yin, X., Zhao, S., Cui, Q., and Zhang, H. (2000). “Predictive Control Applied to Networked Control Systems.” *Frontiers of Model Predictive Control*, T. Zheng, ed., InTech, 67–88.

Zamora, J., and Grossmann, I. (1998). “A global MINLP optimization algorithm for the synthesis of heat exchanger networks with no stream splits.” *Computers & Chemical Engineering*, 22(3), 367–384.

Zandvliet, M. J., Bosgra, O. H., Jansen, J. D., Van den Hof, P. M. J., and Kraaijevanger, J. F. B. M. (2007). “Bang-bang control and singular arcs in reservoir flooding." *Journal of Petroleum Science and Engineering*, 58(1-2), 186–200.

Zhao, W., Varshney, U., Gill, H., and Baheti, K. (2006). "Cyber Physical Systems." *Conference Website*, <http://varma.ece.cmu.edu/CPS/> (Jan. 26, 2014).

Zhong, R., and Sumalee, A. (2008). "Stochastic Cell Transmission Model: Traffic State Estimation under Uncertainties." *Traffic and Transportation Studies*, (1994), 462–478.

ZigBee Alliance. (2014). "Specifications." <http://www.zigbee.org/Specifications.aspx> (Jan. 22, 2014).

Zimmerman, A. (2007). "Parallelized Simulated Annealing for Model Updating in Ad-Hoc Wireless Sensing Networks." *Mobile Data Management, 2007*, Conference and Custom Publishing, 341–345.

Zimmerman, A. T., and Lynch, J. P. (2009). "A parallel simulated annealing architecture for model updating in wireless sensor networks." *IEEE Sensors Journal*, 9(11), 1503–1510.

Zimmerman, A. T., Lynch, J. P., and Ferrese, F. T. (2009). "Market-based computational task assignment within autonomous wireless sensor networks." *IEEE International Conference on Electro/Information Technology*, IEEE, Windsor, Canada, 23–28.

Zimmermann, H. (1980). "OSI Reference Model--The ISO Model of Architecture for Open Systems Interconnection." *Communications, IEEE Transactions on*, 28(4), 425–432.

Zivi, E. L. (2002). "Integrated shipboard power and automation control challenge problem." *Power Engineering Society Summer Meeting, 2002 IEEE*, 325–330 vol.1.