WEB BASED INTEGRATED MULTIPLE FUNCTION CUSTOMER DEMAND AND BUDGET MANAGEMENT SYSTEM

by

SUPUN TIPTIPAKORN

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2008

ACKNOWLEDGEMENTS

I am deeply indebted to Professor Wei-Jen Lee (my academic advisor) and Mrs. Em-orn Tiptipakorn (my stepmother who loves me like her actual own son) for their financial support, opportunity, understanding, patience, and guidance throughout my pursuit of this academic success. They are the real examples of munificence and generosity that I will always look up to and follow for the rest of my life.

I am obliged to Professor Raymond R. Shoults, Professor Kai-Shing Yeung, Professor William E. Dillon, and Professor Hua-Mei Chen for their kindness, guidance, and valuable time to serve as my PhD committee.

I want to dedicate this academic accomplishment to my passed away father, Mr. Praves Tiptipakorn. I wish he knows and is proud of this big step of my success. I adore my mother, Mrs. Neuy Saemuk, and all of my family members for their infinite love and understanding even though I am half the globe away.

I am grateful for the people of the Meditation Center of Texas. Their warm friendship and kind assistance during my PhD study make me enjoy being in Arlington, TX.

To finish, I am grateful for Miss Supichaya Panprasert. Her love and encouragement carry me to achieve this accomplishment.

April 18, 2008

ABSTRACT

WEB BASED INTEGRATED MULTIPLE FUNCTION CUSTOMER DEMAND AND BUDGET MANAGEMENT SYSTEM

Supun Tiptipakorn, PhD.

The University of Texas at Arlington, 2008

Supervising Professor: Dr. Wei-Jen Lee

Traditionally, most end-use residential consumers receive flat rate monthly electricity bills based on load profiling rate calculations from different utilities' consumer classes. Those flat rates do not relate and reflect the true and time-varying costs of electricity supply during actual time of consumptions. Without acknowledging the cost differences, consumers do not have any incentive to adjust their load consumption patterns, and inefficiencies of resource allocations can result. Cheaper-cost off-peak load users can end up paying to offset those expensive peak load users. Additionally, peak load users may still consume their loads without any awareness that their power networks may have already bound to dangerous constraints. Crises and markets failures are often originated from these wholesale and retail market

disconnections. Avoidable investments to increase transmission or generation capacities to satisfy peak loads are unnecessarily spent, and the lead-times of such constructions can take several years.

Providing market signals for end-use consumers and enabling them to adjust their loads, consumers can shift away their loads from high market prices when the power systems are stressed. Demand response program (particularly real-time electricity pricing) can be a proper solution to link both supply wholesale and demand retail markets.

The Federal administrations perceive the necessities of electricity demand response programs and have passed the Energy Policy Act (EPAct) 2005 to provide supporting infrastructures and technologies for the demand response programs for all classes of consumers. Accordingly, this research realizes the coming developments and the vitality of the demand response for effective and efficient electric deregulations. The simple yet effective consumer-centered (contrast to utility-centered) load control strategies (Steps of Temperatures and Pricing Naming) are introduced and integrated to the web based integrated multiple function customer demand and budget management system. The proposed system can serve as a tool to assist end-use residential consumers for their load and budget managements in real-time pricing environments, like Illinois (the state where 4.5 million consumers can participate in such program). Other features can further add-on to enhance the proposed system to better reach end-use consumers (e.g., home security and surveillance system).

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background

Many previous crises in the electric power system, which widespread to millions of people, and billions of dollars were lost, were originated from lack of linkage between wholesale and retail markets [1], [2], and [3].

At present, most residential customers receive monthly electric bill at a static rate for the electricity consumed. That constant rate, however, is not directly related to the true costs of electricity provided during consumers' actual consumptions [4]. When electricity supply is deficient, consumers have no motivation to reduce their electricity usages if their rate of electricity is not affected by the dynamic price changes, especially during the hot summer day when heavy air conditioning loads are consumed (a case of the California Crisis).

Fig. 1.1 and 1.2 illustrate supply short and demand imbalances in California 2000-2001. Supply shortage created highly volatile electricity market in California throughout year 2000. The demand inelasticity forced electric utilities to pay electricity almost at any price to meet the demand. Between June 2000 and April 2001, Pacific Gas and Electric Company spent $9 Billion in excess of revenues to buy power for its customers. The company filed chapter 11 on April 6, 2001. For residential customers in

San Diego area, the energy portion of the electricity bill jumped more than six times from approximately 3 cents per kilowatt hours in Jan 2000 to 21 cents per kilowatt hours in September 2000 (Fig. 1.2). Total losses in the State of California are estimated to be fourteen billion dollars.

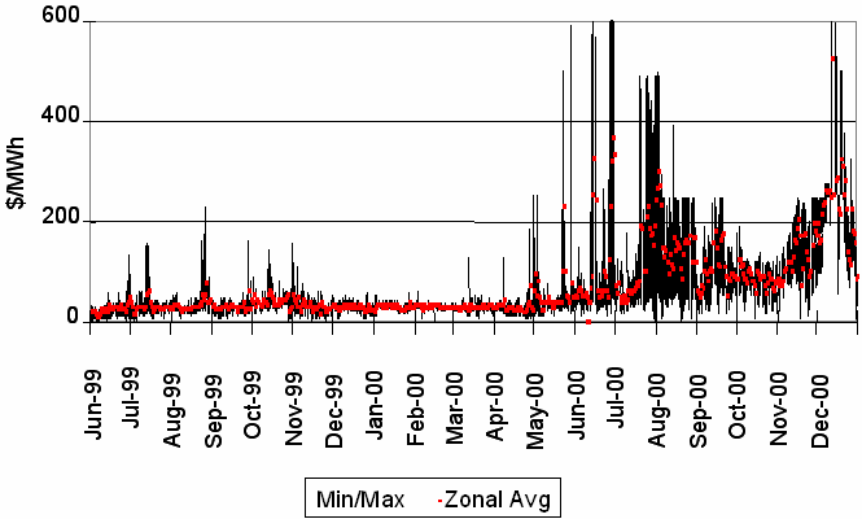


Figure 1.1 California Day-Ahead electricity prices (PX – Southern Zone) [5]

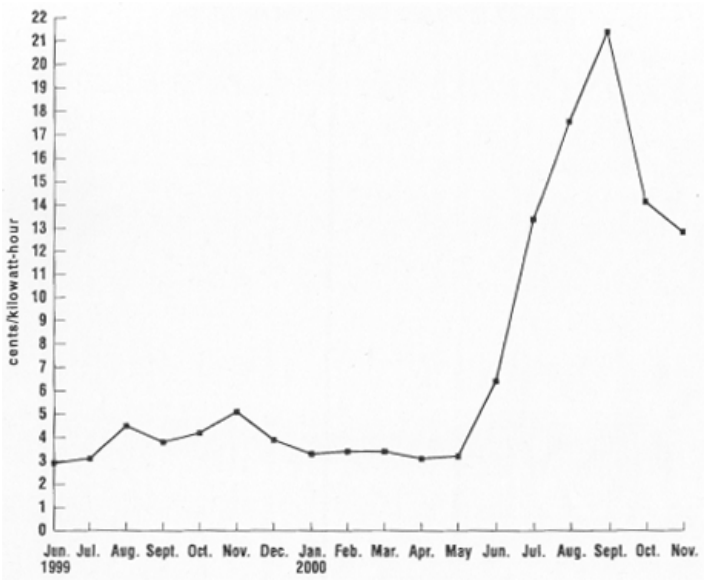


Figure 1.2 San Diego's residential energy portion of the electricity bill [6]

The disconnection between wholesale and retail electricity markets is also the fundamental cause of the first prominent price spike (sharp increase of price in a short period) in the United States, the Midwestern price spike in June 24 and 25, 1998. The wholesale electricity prices reached the unprecedented of $7,500 per MWH for a few transactions (Fig. 1.3). Another even greater price spike of approximately $10,000 per MWH in Midwest was also noticed in July 29 and 30, 1999 [3].

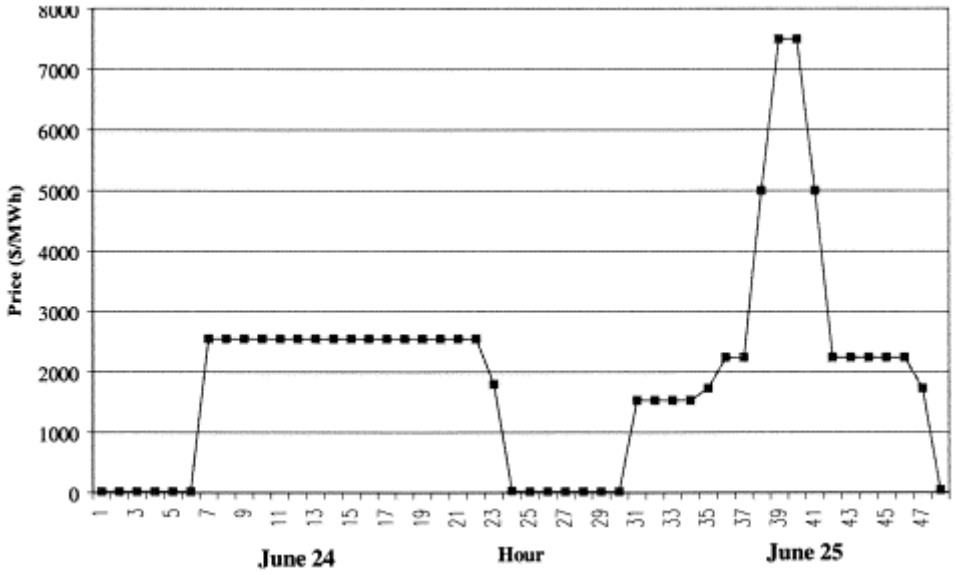


Figure 1.3 Hourly trade prices on June 24 and 25, 1998, in the Mid-America Interconnected Network (MAIN) [3]

Another case in another region, in summer 2000, the New England ISO accepted bids under a contract with the New York ISO that resulted in an energy price of $6,000/MWH for four hours (Fig. 1.4) [7].

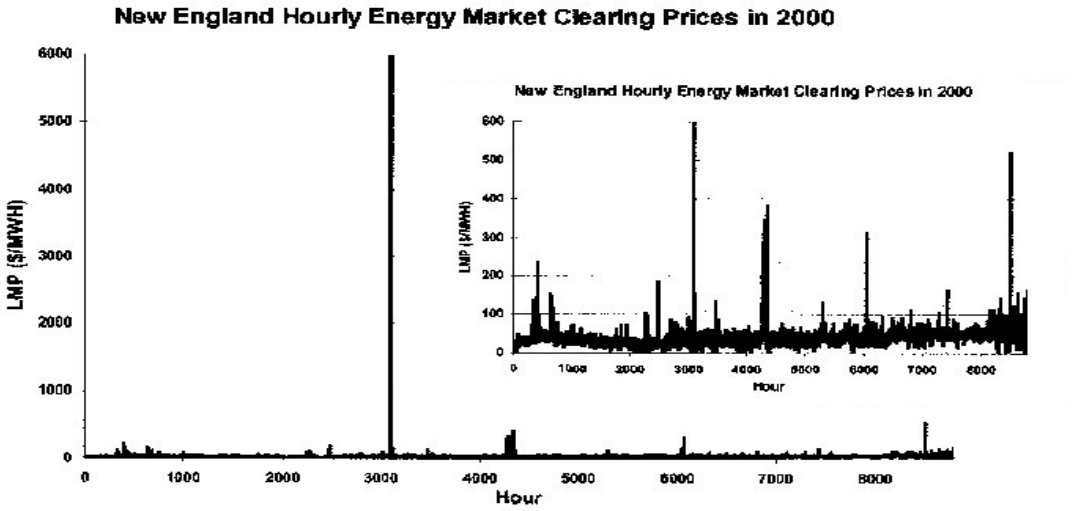


Figure 1.4 New England ISO reached $6,000/MWH energy price
for 4 hours on May 8, 2000 [7]

Market failures and price spikes are very costly and severe and can occur in any regional reliability councils. Thus, it is important to understand their economics mechanism behind. To begin, the simple examples (Fig. 1.5 and 1.6) [3] and [8] will be illustrated below:

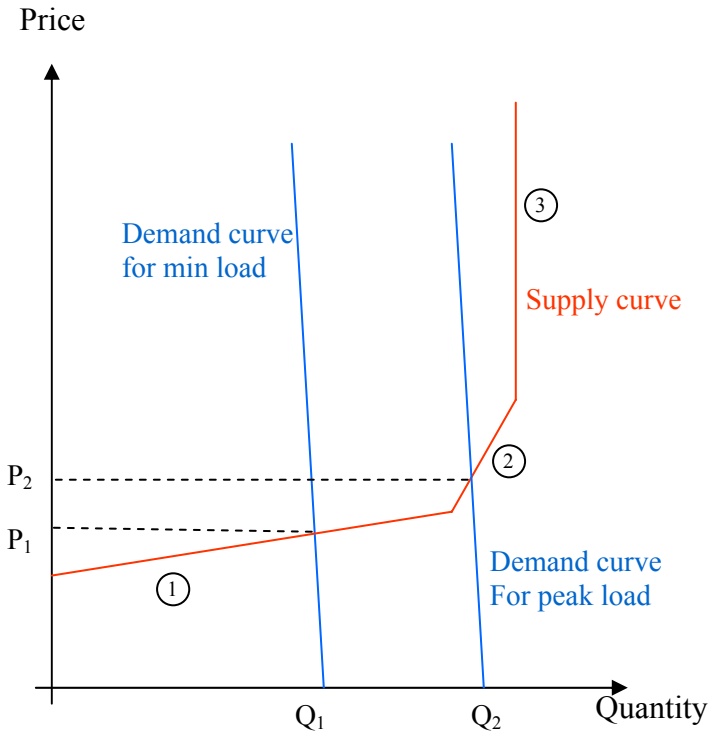


Figure 1.5 Mechanism leading to market failures and price spikes
– sufficient generation capacity (P_1: price at min load, P_2: price at peak load)

A normal aggregate electric generation supply curve can approximately characterized to three piecewise linear sections (Fig. 1.5). The first section, almost a horizontal line represents generation supplies from low cost bulk generating units. High cost peak units do not have to be committed for the increased demands within this region. The incremental costs for generation are low and the supply curve is near perfectly elastic. Large increase in generation supply, yet, results in small increase in price. The second section, having a stepper slope, represents generation supplies from higher cost, peak units that are committed infrequently to satisfy the increased demands. Although generation supplies in the section 2 of the supply curve are adequate to satisfy

the increased demands, the generation capacity is tight and close to its limit. Higher incremental costs are resulted from these higher cost peak generating units that have to be committed to the system. The third section, almost a vertical line, represents all existing generation supply capacity reaching its limit. The incremental costs are extremely high and the supply curve is near perfectly inelastic. Small increase in generation supply results in extremely high price.

In Fig 1.5, when aggregate demands increase from minimum generation period to higher generation periods, the demand function Q_1 moves horizontally rightward to Q_2. P_1 (min price) and P_2 (higher price) are the market equilibrium prices that are the intersections between the corresponding aggregate supply and demand curves.

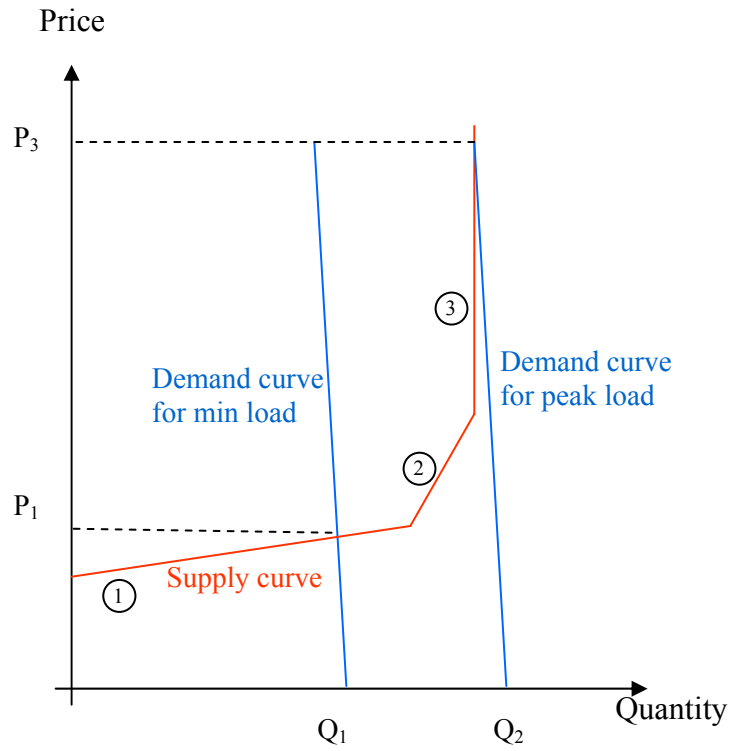


Figure 1.6 Mechanism leading to market failures and price spikes
– insufficient generation capacity (P_1: Price at min load, P_3: Price at peak load)

In Fig. 1.6, price spikes P_3 occur when all existing installed generation capacity is near its limit (section 3 of the supply curve). Regardless of how high the prices of the buying bids, the generating units can no longer offer any further supply. The limit supply capacity can be resulted from, for example, unit maintenance, unexpected high demand in hot summer, scarce water supply for hydro plants from drought, or market power gaming.

Market power gaming can cause high prices in electricity or price spikes. Power producers can dishonestly increase profit by market power gaming and intentional withholding their generating capacity to limit the supply. A real case of California day-ahead energy market on September 28, 1999 [9] is discussed below to illustrated an example of price spikes from market power gaming. At hour 20^{th} transaction, the market clearing price jumped to above $200/MWH from a normal less than $50/MWH daily average.

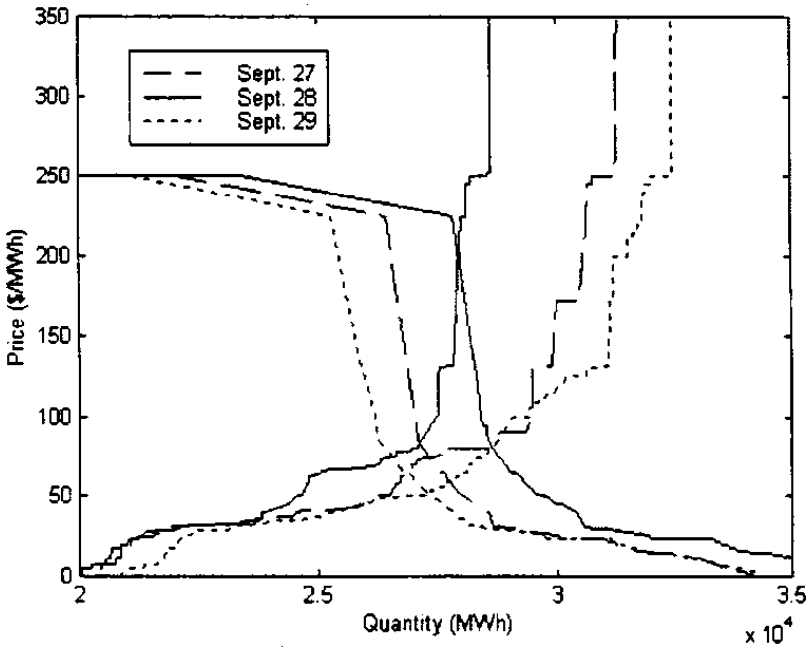


Figure 1.7 Aggregated bids for September 27–29, 1999 at hour 20th [9]

From Fig. 1.7, the market clearing demands between September 27th and 29th at hour 20 were nearly identical (28,835 MWH for September 27th, 29,410 MWH for September 28th, and 28,250 MWH for September 29th). The aggregated supply offer on September 28th (the day that having price spikes), however, was much less than the aggregated supply curves of September 27th (the day before having price spikes) and September 29th (the day after having price spikes). The corresponding price at hour 20th of September 28th was higher than $200/MWH relative to the prices of approximately $80/MWH at the same hour 20th on September 27th and 29th. Assessing these indications, it can be presumed that the market power gaming in intentionally withholding the generating capacity on September 28th was occurred.

1.1.1 Definition of Demand Response

U.S. Department of Energy (DOE) defines the demand response as follows:

Changes in electric usage by end-use customers from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized.

From this definition, demand response can be classified into 2 categories: first, time-based and second, incentive based.

The time-based consists of time-of-use (TOU), critical-peak, and real-time pricings. The incentive-based consists of direct load control, interruptible, demand buyback, emergency demand response, capacity market, and ancillary services market.

The definitions and developments for each type are detailed in [10].

1.1.2 Types of Time-Based Demand Response

In this section, the time-based demand response is detailed since the enactment of Energy Policy Act 2005 centers its importance, and the time-based demand response (particularly, real-time pricing) is the core of this dissertation.

The definitions of various types of time-based pricings defined in the Energy Policy Act 2005 Section 1252 (a) (14) (B) [11] are quoted as follows:

‘‘(i) time-of-use pricing whereby electricity prices are set for a specific time period on an advance or forward basis, typically not changing more often than twice a year, based on the utility’s cost of generating and/or purchasing such electricity at the wholesale level for the benefit of the consumer. Prices paid for energy consumed during

these periods shall be pre-established and known to consumers in advance of such consumption, allowing them to vary their demand and usage in response to such prices and manage their energy costs by shifting usage to a lower cost period or reducing their consumption overall;

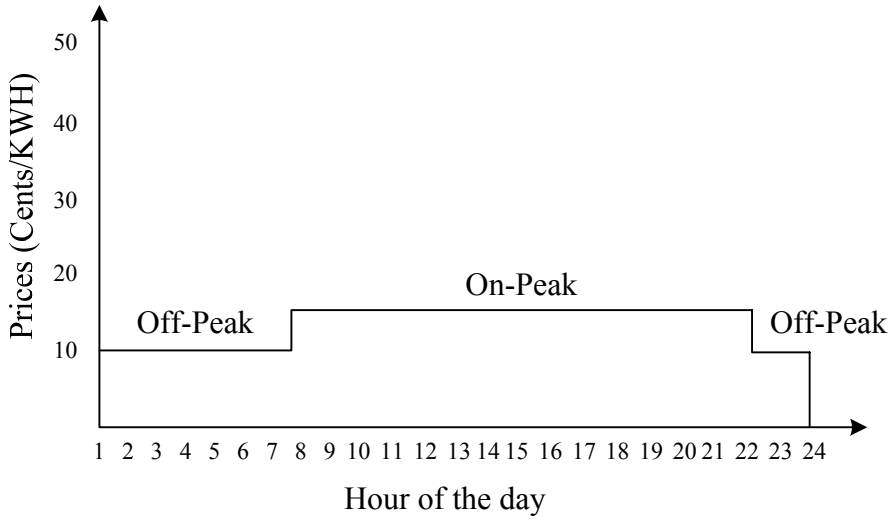


Figure 1.8 Time-of-use (TOU) pricing [10]

‘‘(ii) critical peak pricing whereby time-of-use prices are in effect except for certain peak days, when prices may reflect the costs of generating and/or purchasing electricity at the wholesale level and when consumers may receive additional discounts for reducing peak period energy consumption;

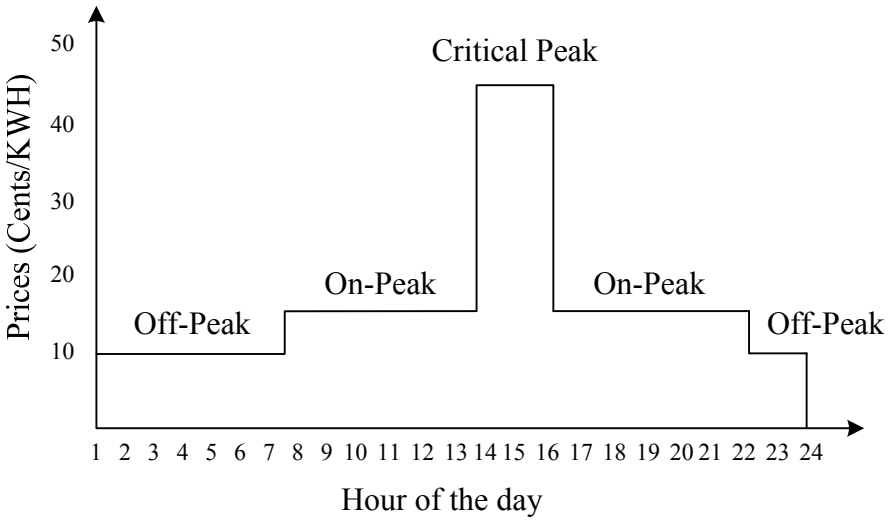


Figure 1.9 Critical peak pricing [10]

‘‘(iii) real-time pricing whereby electricity prices are set for a specific time period on an advanced or forward basis, reflecting the utility’s cost of generating and/or purchasing electricity at the wholesale level, and may change as often as hourly;

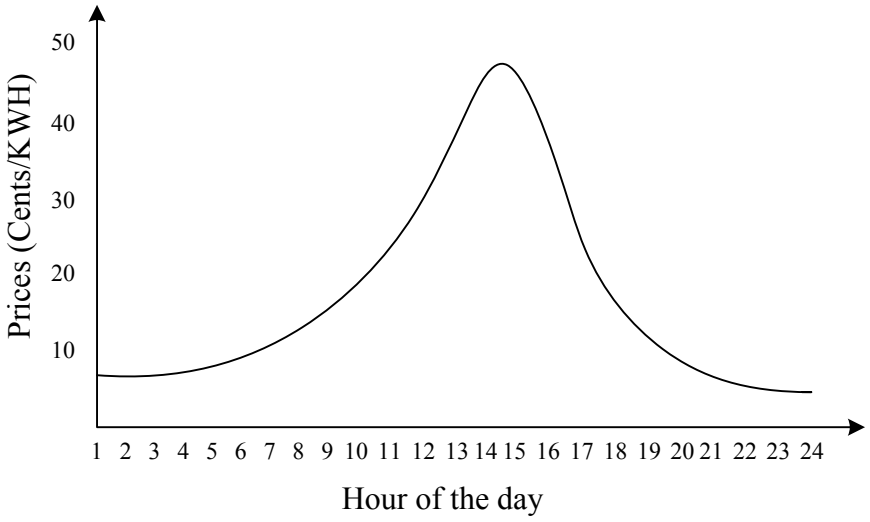


Figure 1.10 Real-time pricing [10]

Time-based pricings for retail customers are mechanisms to allow the customers to adjust their load consumptions according to the price signals they perceive. They can reduce or defer their load consumptions during the high electricity price periods and, conversely, can take advantage of the low electricity price periods by increasing their load consumptions.

1.1.3 Mechanism of Demand Response to Alleviate Price Spikes [3]

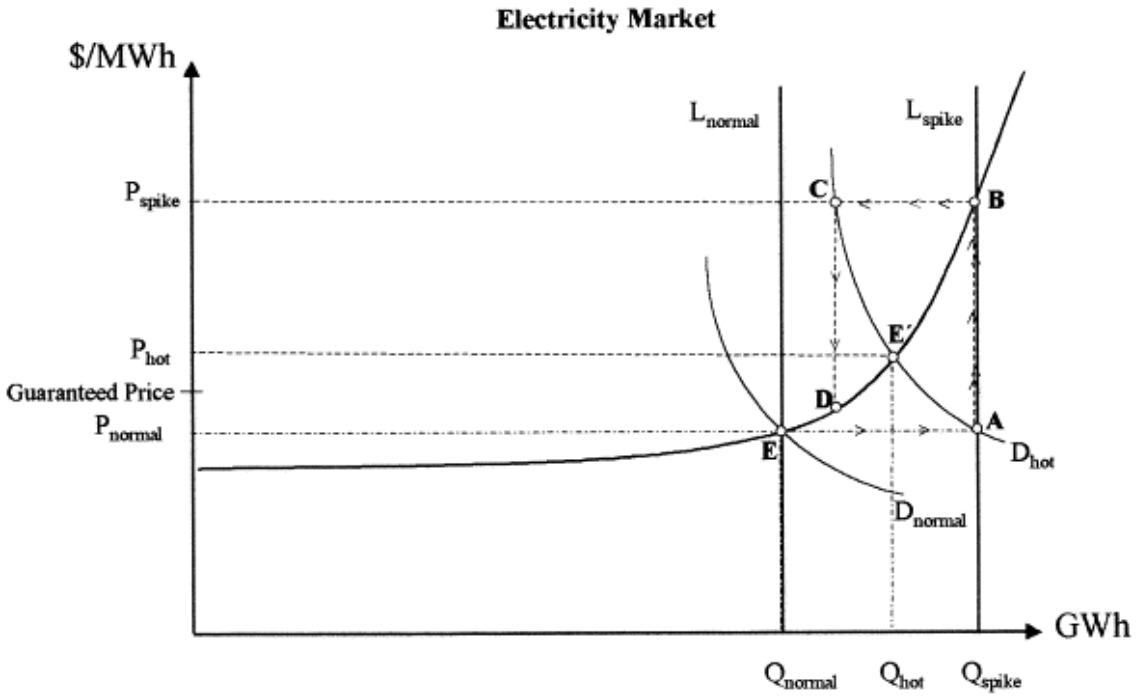


Figure 1.11 Mechanism of demand response to alleviate price spikes [3]

When there is disconnection between wholesale and retail of electricity markets, an aggregate demand can be represented as a line nearly vertical to the quantity axis (near perfectly inelastic demand). The equilibrium price is the price that is the

intersection between the aggregated demand and supply curves, from Fig. 1.11, P_{normal} at point E. The aggregated demand line will move rightward, for example, from increases air conditioning load consumption during hot weather. The higher demand (higher than supplied) moves aggregated demand curve rightward from Q_{normal} to Q_{spike}. The excess demand pressures the commitment of high cost generation units to the new equilibrium point B and creates a price spike, P_{spike} discussed earlier.

If some degree of connection between the wholesale and retail electricity markets exists (some loads are traded on spot based market, in other words, having demand response), the aggregate demand curve will be a slant downward curve, instead of a vertical line to the quantity axis. The increase in price from generation units at P_{spike} leads the price-conscious customers to reduce their demand consumptions and move the quantity to point C in the new slant aggregated demand curve. At point C, the excess supply now occurs and, in turn, drives the price downward to point D in the supply curve. The price and demand-supply correction and re-correction processes persist until reaching the final equilibrium point at E′. The final price is P_{hot}, which is greatly less than P_{spike} where there is no demand response exists.

The FERC's Staff Report [10] and the Department of Energy's Report to the US Congress [12] describe the benefits, barriers, and recommendations as the frameworks to encourage the implementation of this demand response, which can be summarized in the following sections.

1.1.4 Benefits of Demand Response

The primary benefit of demand response is the enhancement of effectiveness and efficiency in utilizing the system's overall resources both long-term and short-term. An effective demand response program can reduce or shift their electricity consumptions away from the peak periods of near capacity limits and high marginal generating costs, the constrained or stressed power systems can be mitigated. The transmission networks, consequently, are not in at risk of overload, which can avoid forced outages or blackouts. Market failures and crises, such as California meltdown in 2000 and 2001 can be prevented. The expensive-to-run low efficiency peak generators do not have to be committed. Environmental benefit can also result since these low efficiency generators usually have highly polluted emission.

In addition to reduce the potential cause of market failures, avoidable long term investments in constructing new power plants and transmission lines to meet the always unsatisfied peak period demand can also be achieved. As we know, to increase capacities to relieve the system constraints through physical infrastructure improvement can take several years of lead-time. Demand response can be an immediate solution with a much shorter timeframe to execute. Investment resources to build new transmission networks or new power plants can be conserved, and the savings on the aggregate supply side maybe eventually passed onto the end retail consumers.

Generators can no longer charge extremely high rates (the case of inelastic disconnect retail market where consumers have to yield to pay those high rates).

Demand response can also lessen market power gaming preventing generators to dishonestly withheld supply to raise prices.

Given the ability for consumers to say "not at that price" [13], they can be more in-control of their energy usages and electricity budget managements. Conscious and price responsive consumers can save since the electricity bills from the market-based pricing usually (but not guaranteed) are lower than from the traditional flat average rates.

For consumers' perspective, after seeing the traditional flat rate electric bills, they can be upset by the high rate after the electricity was already consumed a month ago (no one will ever purchase and use any merchandize without knowing the price first, but why we treat the merchandize of electricity differently) [14].

Additionally, charging consumers with the traditional monthly flat rate based on the average load-profiling rate calculation of consumer classes (regardless of consumers usage behaviors during the days) can result in, for example, the consumers with high load consumptions during cheap off-peak evening are overcharged to offset for the consumers with high load consumptions during expensive peak hours. This approach is not fair and should not be the customer service of the 21st century.

1.1.5 Energy Policy Act (EPAct) 2005

The US Government perceives the necessity of the demand response from the end-use residential consumers as stated in the Energy Policy Act (EPAct) 2005 Sections 1252 (a) (14) (A), 1252 (d), 1252 (e), and 1252 (f) [11]:

"(a) (14) (A)…each electric utility shall offer each of its customer classes, and provide individual customers upon customer request, a time-based rate schedule under which the rate charged by the electric utility varies during different time periods and reflects the variance, if any, in the utility's costs of generating and purchasing electricity at the wholesale level. The time-based rate schedule shall enable the electric consumer to manage energy use and cost through advanced metering and communications technology.

''(d) DEMAND RESPONSE.—The Secretary shall be responsible for—

''(1) educating consumers on the availability, advantages, and benefits of advanced metering and communications technologies, including the funding of demonstration or pilot projects;

''(2) working with States, utilities, other energy providers and advanced metering and communications experts to identify and address barriers to the adoption of demand response programs;

(e) DEMAND RESPONSE AND REGIONAL COORDINATION.—

(1) IN GENERAL.—It is the policy of the United States to encourage States to coordinate, on a regional basis, State energy policies to provide reliable and affordable demand response services to the public.

Section 1252 (f) Federal Encouragement of Demand Response Devices

It is the policy of the United States that time-based pricing and other forms of demand response, whereby electricity customers are provided with electricity price signals and the ability to benefit by responding to them, shall be encouraged, the

deployment of such technology and devices that enable electricity customers to participate in such pricing and demand response systems shall be facilitated, and unnecessary barriers to demand response participation in energy, capacity and ancillary service markets shall be eliminated.

The deployments of technologies and infrastructures to enable the residential consumers to participate in demand response are apparently forthcoming.

1.1.6 Recent Experiences of Selected Demand Response Programs

In addition to the following discussion summarized from [13], [15], [16], additional detailed information can be found in the utilities' websites referenced within the sections:

a) Georgia Power

Georgia Power [17], a subsidiary of the Southern Company, offered a pilot real-time pricing program (consumers' choices of hour ahead or day-ahead notifications) to 1,639 large industrial consumers having peak demand of 5,000 MW. When wholesale market was constrained during high-priced hours, participants were able to reduce demand at least 750MW. The reductions in peak demand were up to 17 percent on critical days. The costs on utilizing the expensive peak generating units were reduced, and the savings were passed to its consumers.

A rural textile mill participating in Georgia Power's real-time pricing program reported that during the electricity high price periods, the mill reduced its electricity purchases from Georgia Power and utilized its on-site generator. The savings were approximately $1 million per year.

b) NYISO

Three demand response programs are currently offered by New York Independent System Operator (NYISO) [18]:

1) The Emergency Demand Response Program (EDRP) is a financial incentive for large customers to participate in a voluntary consumption curtailment program (2-hour-ahead notification) without penalty of not participating.
2) The Special Case Resources (SCR) is a financial incentive consumption curtailment program (2-hour-ahead notification) similar to EDRP for electricity consumers larger than 100 kW. During market constrained, SCR consumers will be called first, followed by EDRP, if only additional curtailments are required.
3) The Day Ahead Demand Response Program (DADRP) allows large consumers to submit bid for their load reduction (between $50 and $1,000 per MWH) on the following day consumption in New York's day-ahead wholesale electricity market.

During recorded high price of hot summer 2001, load reductions were higher than 25 MW with estimated benefits from this reduction of $1.5 million dollars. The demand response programs were also credited for NYISO enhanced networks' stability and reliability.

c) Gulf Power

Gulf Power [19], a regulated Florida utility, offered voluntary time-of-used (TOU) electricity rates for approximately 3,200 end-use residential customers. The rates were classified into 3 tiers: peak, off-peak, and mid-peak. One additional infrequent very high rate (critical peak price) was in effect with advanced notification when supply was tight.

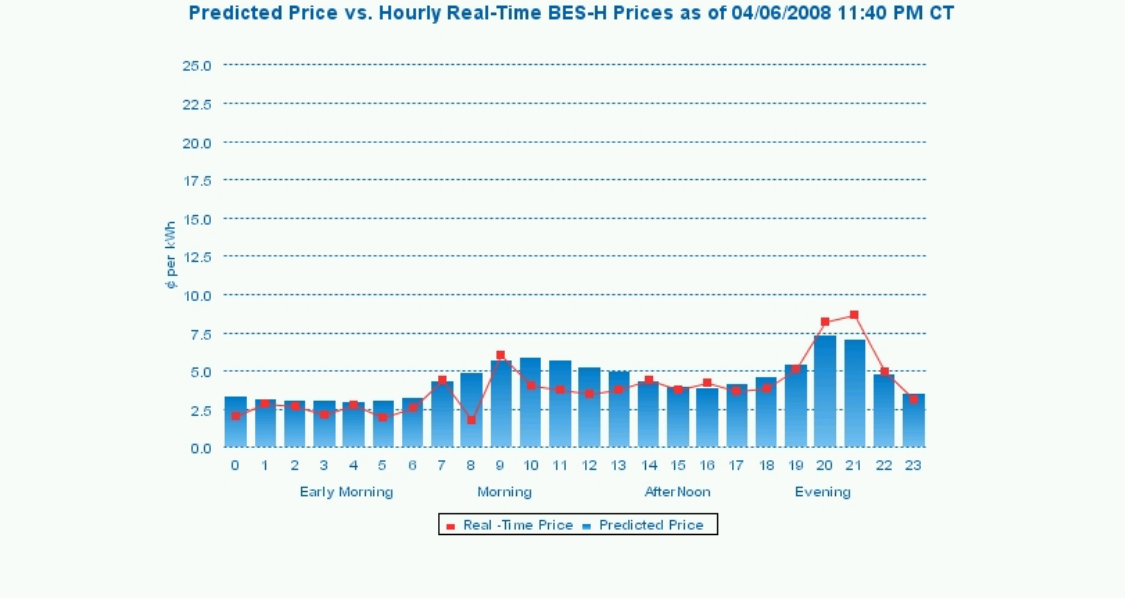
Participants paid electricity bills approximately 11% less than those who did not participate. Some customers reported their saving nearly $600 per year when they shifted their usages of appliances to off peak hours. Success of the demand response program of Gulf Power can be attributed to the great rate difference of more than sevenfold between high rates and low rates. Customers had high incentive to participate in the program to shift their electricity usages for savings.

d) Illinois

During 2003 to 2006, CNT Energy (a nonprofit organization and a division of the Center for Neighborhood Technology) [20] operated the Energy-Smart Pricing Plan (ESPP), a pilot program of day-ahead electricity pricing to end-use residential customers. Participants were notified of the following day electricity price via a website, or a toll free phone number. The customers were responsive to the expected high prices of electricity, and the savings were approximately 20%.

In early 2007 Illinois Legislature enacted a bill requiring the major Illinois utilities, including the Ameren Illinois Utilities [21] and Commonwealth Edison (ComEd) [22], to cooperate with third party program administrators to promote the

voluntary real-time hourly market-based price of electricity to statewide 4.5 million end-use residential consumers. Ameren Illinois Utilities contracted the CNT Energy as its program administrator and introduced Power Smart Pricing [23]. Ameren's customers can check the estimated day-ahead and the real-time prices of electricity by visiting the website [24], or a toll free number. Comverge [25] was selected as the program administrator for ComEd and introduced Basic Electric Service-Hourly Energy Pricing (BES-H) to its residential consumers. The ComEd's estimated day-ahead and real-time prices are posted on the website [26] as in Fig. 1.12, or obtained by a toll free number. If the prices are higher than the customers preset price, customers can choose to receive e-mail, voicemail, or text message alerts. The results on savings since the program started from May 2007 show the participants can reduce their electricity bills on an average of 16% compared with the traditional bills on the standard flat residential rates.



Hour Ending	0	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11
Predicted Price	3.3	3.1	3.0	2.9	2.9	2.9	3.2	4.2	4.8	5.7	5.8	5.6	5.2	4.9	4.2	3.9	3.8	4.1	4.5	5.4	7.2	7.0	4.7	3.4
Actual Price	1.9	2.8	2.6	2.0	2.7	1.9	2.5	4.4	1.8	6.0	4.0	3.7	3.4	3.7	4.3	3.7	4.2	3.6	3.8	5.0	8.1	8.6	4.8	3.1

Figure 1.12 ComEd's website informing its residential consumers predicted and hourly real-time prices [22]

1.1.7 Demand Response Resource Contributions

The regional reliability councils defined by North American Electric Reliability Council (NERC) is illustrated in Fig. 1.13 to show demand response resource contributions.

a) Electric Reliability Council of Texas, Inc. (ERCOT)

b) Florida Reliability Coordinating Council (FRCC)

c) Midwest Reliability Organization (MRO)

d) Northeast Power Coordinating Council (NPCC)

e) ReliabilityFirst Corporation (RFC)

f) SERC Reliability Corporation (SERC)

g) Southwest Power Pool, Inc. (SPP)

h) Western Electricity Coordinating Council (WECC)

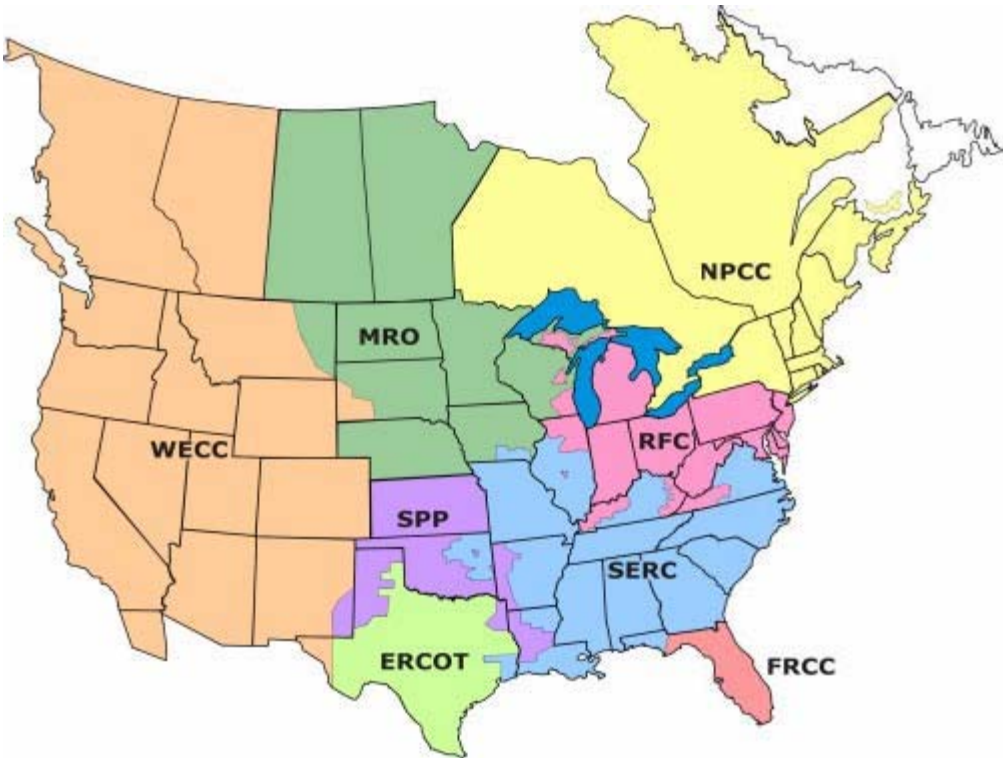
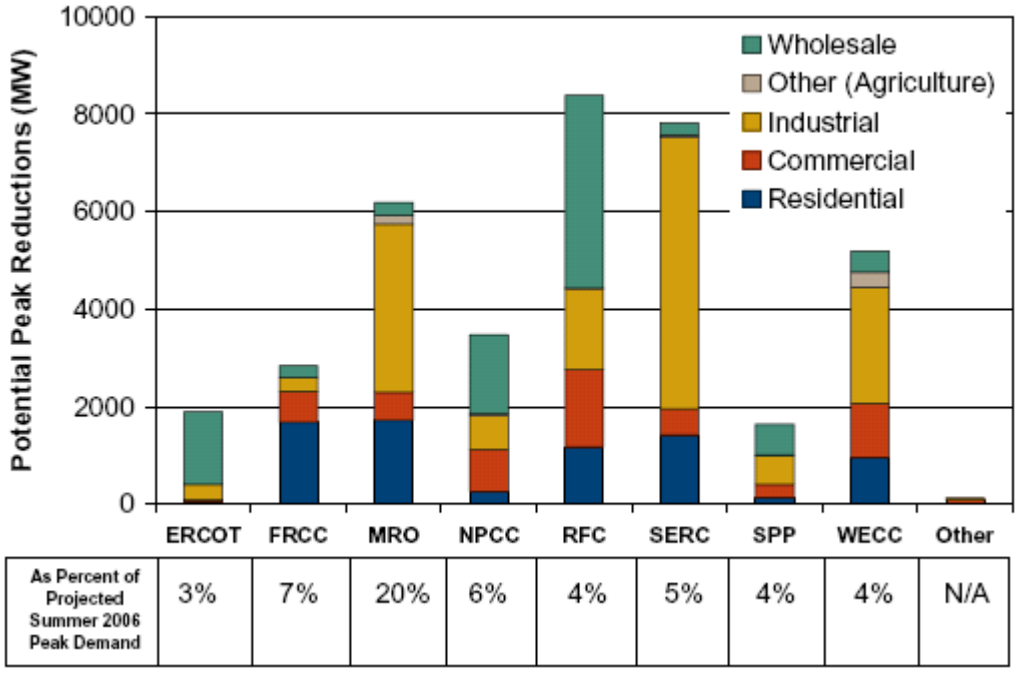


Figure 1.13 NERC Region Map [10]

Demand response resource contribution is estimated at 37,552MW nationally or 5% of total electricity demand in summer 2006. Fig. 1.14 illustrates the contributions by NERC regions and customer types which vary in the level of 3% to 7% (except MRO with the highest contribution of 20%). Residential customers account for approximately 20% of the national contributions, and some regions have contributions more than 1,000 MW, for example, FRCC, MRO, RFC, SERC, and WECC. Demand

response contributions from ERCOT account approximately 3% (less than 2,000MW) and largely from wholesale and industrial transactions.



As Percent of Projected Summer 2006 Peak Demand	3%	7%	20%	6%	4%	5%	4%	4%	N/A

Figure 1.14 Existing demand response resource contributions by NERC region and customer types [10]

Fig 1.15 classifies the resource contributions corresponding to various types of demand response programs. Interruptible program accounts approximately 7,504MW (27% of national level), mostly contributed from industrial customers (84% of total of interruptible program). 84% of total direct load control program is contributed from residential customers (more than 5,000MW). An example of direct load control program for residential customers is that a utility curtails or cycles down the consumptions of its consumers' central air conditioner, heater, or water heater. Time-

based tariffs such as real-time, critical peak, or time-of-use pricings are not the significant resources of demand response contributions.

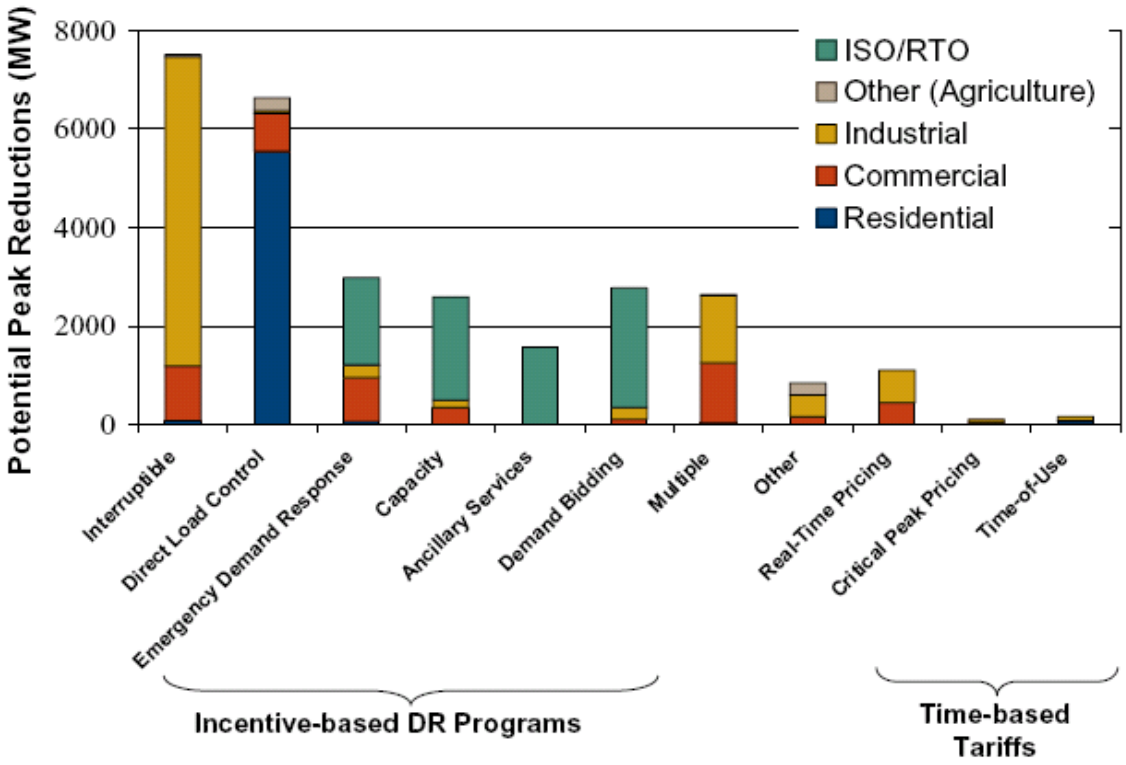


Figure 1.15 Resource potential of various types of demand response programs [10]

Even though the contributions of demand response in current deregulation are not significant, some researchers [3], [27] and [28] state that even with small participation (less than 10%) on demand response can effectively alleviate the price spike on the load-duration curve.

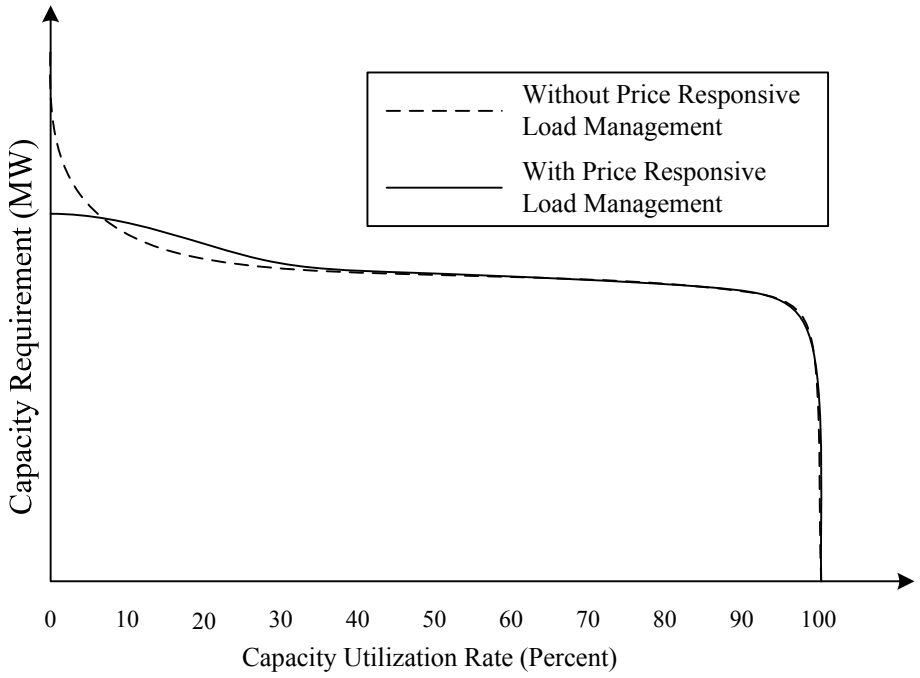


Figure 1.16 Small participation on demand respond can alleviate the price spike [27]

1.1.8 Barriers in Implementing Demand Response Programs

Although the numerous benefits of demand response are identified, the penetration and implementation of the program are not extensive as discussed in section 1.1.7. Many barriers and challenges in implementing the demand response program will be explained as follows:

To successfully implement the demand response program requires supported infrastructures to link wholesale and retail markets: for example, advanced metering and other enabling technologies. Market-based pricing demand response needs the meters

that have the ability to measure, record, store, and communicate household' electricity consumptions by, for example, hourly or TOU periods, in which the traditional meters cannot perform. The utilities also need communication channels to inform time-based prices to their participating customers, so that their customers can response and shift the load consumptions accordingly. Also, the technologies to assist consumers to decide their load consumptions corresponding to the changing electricity prices are limited.

Utilities are reluctant to invest in these long-term infrastructures while the immediate monetary gains to implement the demand response programs are not apparent. Besides, utilities obtain revenues based on peak MW and usage MWH of their consumers. From their perspectives, implementing demand response can cause their short term revenues to be lower and dissatisfy their stock or stake holders.

Consumers' inertia and their not willingness to change can also impede demand response implementations. They are not well-informed before regarding the differences of the electricity costs during the day and the benefits they can attain if they choose to shift their load consumptions. To instill the habits of price consciousness and responsiveness cannot be accomplished instantaneously. They can resist to changes if the program is not simple and requires too much effort.

Some peak load consumers, who benefits from the traditional flat rates, maybe reluctant to participate in market-based pricing demand response because they can end up paying higher.

Price caps and some state regulations that prevent their residents to be exposed to time-varying market-based electricity rates can also impede the implementations of the demand response.

1.1.9 Deficiencies in Current Electric Deregulations

Current electricity deregulated environment has some deficiencies as follows:

a. In perspective, present schemes are focused more on utility than customers; for example, direct load control sheds/disconnects load for network security or from generation shortage, not for consumers' comfort; AMR (Automatic Meter Reading) is used for utility meter reading but not for providing consumers' power consumption information for their load management decision making.

b. Most utility demand response programs do not have dynamic price information available to all residential consumers to adjust their demand behaviors (merely some large industrial consumers can participate in). Without participations from all consumer classes, the electric deregulations cannot be fully accomplished.

c. The demand response programs for residential consumers are largely the time-of-use electricity pricing and direct load control. The implementation in real-time pricing (the most effective approach to reflect the true costs of generating supply) is limited.

d. The necessary infrastructures and technologies to support residential consumers' demand response have not been fully deployed.

e. For Texas, ERCOT (The Electric Reliability Council of Texas) will change from Zonal pricing to Nodal pricing (Texas Nodal) in January 2009 and will have more

than 4,000 nodes [29]. Different locations will have different electricity real-time prices. To charge all residential customers at a monthly static rate is very inefficient and obsolete.

1.2 Research Motivations and Objectives

From the necessities of electricity demand response program (particularly real-time pricing) and potential developments of its implementation, we need a tool to enable the end-use residential consumers for an effective demand response. This tool has to be simple and provides essential information (such as real-time pricing, temperatures) for the consumers to properly and easily adjust their load consumptions for savings. The tool should have an approach to control the loads corresponding to the consumers' commands. Unlike the utility-centered perspective (such as direct load control strategy, which sheds loads when the electrical network is stressed regardless of the consumers' discomfort), the tool has to be consumer-centered, in other words, providing consumers' control preferences according to the real-time prices of electricity. Besides, the tool should effectively balance the consumers' comforts relative to their savings. All discussed topics are the theme and mission of this dissertation.

1.3 Synopsis of Chapters

This dissertation is organized as follows:

Chapter 1 provides background, benefits, and necessities of demand response programs to link wholesale and retail power markets for effective and efficient electricity deregulation. The behind economics mechanisms from consumers to prevent or mitigate price spikes, which can lead to market failures, are explained. A real case of

market power gaming as a cause of price spikes in California market is presented. The types of demand response program in the US are defined according to Energy Policy Act 2005. Previous utilities' experiences of demand response in US electricity markets are described follows by the demand response resource contributions from various US regional reliability councils and types of consumer classes. The challenges and barriers impeding the US demand response programs are also examined. The objectives and motivations of this dissertation to develop a tool to serve end-use residential consumers to enhance demand response in real-time electricity pricing environment are explained.

In chapter 2, the Steps of Temperatures load control strategy to assist those end-use residential consumers for their load controls of air conditioner/heater and water heater is proposed. Simulations are executed and the performances of the strategy are validated.

The well-known Price Naming from on-line reservation is applied as one of this dissertation's load control strategy and is introduced in chapter 3 to help the end-use residential consumers in controlling loads such as washer/dryer and dish washer in real-time electricity pricing environments. The electricity cost savings comparing with the waiting times from deferring load usages to later cheaper off-peak periods are simulated and presented.

In chapter 4, the proposed load control strategies (i.e., the Steps of Temperatures from chapter 2 and the Price Naming from chapter 3) are implemented in a web based prototype. The MATLAB graphic user (GUI) interface is programmed to accomplish these tasks. For demonstration purpose, the well-established home automation of X10

power line carrier protocol for load control is adopted. The background of X10 home automation is presented therein. The 300 MHz radio frequency is also implemented as an alternate protocol in the GUI. The real-time information of prices and outdoor temperatures are exhibited to the consumers through GUI. Other features that can be further added-on (for example, surveillance camera or home security) are discussed as well.

Chapter 5 is the conclusion and summary of the dissertation's research contributions. The possible further researches and studies are outlined to conclude the dissertation.

CHAPTER 2

PROPOSED LOAD CONTROL STRATEGY: STEPS OF TEMPERATURES

To control load consumptions, loads first have to be correctly categorized. [27] differentiates residential electric loads into 3 types: re-schedulable usage loads, re-schedulable usage and service loads, and non-reschedulable usage and service loads. The re-schedulable usage loads are the loads that have thermal inertia (e.g., air conditioner/heater). These loads can be, to some degree, deferred their usages to the subsequent timeframes. The re-schedulable usage and service loads are the loads that can be deferred to any other timeframes (e.g., dishwasher, cloth washer and dryer). Lastly, the non-reschedulable usage and service loads are the loads that cannot be deferred at all (e.g., TV, lights, refrigerator).

In this chapter, the experiments on the load control strategy of air conditioner/heater and water heater loads are primarily focused since these loads together account the highest of total household's energy consumption [30] and [31]. Unlike the utility-centered direct load control strategy such as described in [32], which sheds loads when the electrical networks are stressed regardless of the consumers' discomfort, the proposed strategy provides consumers' control preferences according to the real-time prices of electricity. The algorithms previously discussed in [4], [27], [33], [34], and [35] are extended further in this dissertation and the results will be illustrated

in MATLAB [36] simulations using ERCOT actual real-time pricing [29] and outdoor temperature data [37]. The proposed algorithm is effective, yet simple enough for a real-life use of residential consumers.

2.1 Air Conditioner/Heater Load Control

Though some are using adjustable thermostat controllers, the normal practice of most residential consumers in operating air conditioner/heater is to always set the thermostat at a constant temperature setting point. In real-time electricity pricing environment, this practice is inefficient and costly in that the air conditioner/heater still operates at the same setting point even the price of electricity is high. Conversely, when the price is low, this practice does not take advantage of that low price to operate the air conditioner/heater the coldest/hottest allowable to reserve the thermal energy for the subsequent periods.

As shown in Eq. 2.1, the model and parameters discussed in [4], [33], and [34] will be the basis to simulate the indoor temperature of the next timeframe.

$$T_{i+1}^{in} = \varepsilon T_i^{in} + (1-\varepsilon)(T_i^{out} \pm \eta q_i / A) \tag{2.1}$$

("+" for heating and "–" for cooling)

where T_i^{out} - hourly real time outdoor temperature (°F)

T_0 - initial temperature (°F), default value = 77

η - coefficient of performance (COP), default value = 2.5

$q_{\max}$ - maximum power output of AC (KW), default value = 3.5

ε - system inertia, default value = 0.93

A - thermal conductivity (KW/°F), default value = 0.14

To minimize the discomfort of the consumers, air conditioner/heater are operated within the ASHRAE comfort zones [38]. Comfort zones are seasonal: the summer zone is separated from the winter zone. Factors such as dew point temperature and humidity ratio can affect the operative temperature. As shown in Fig. 2.1, both zones have bands of approximately six degrees Fahrenheit.

2.1.1 Real-Time Outdoor Temperature Data

Real-time local temperature data for consumers' demand response is acquired from [37]. Fig. 2.2 shows the outdoor Dallas/Fort Worth temperature data on a hot summer day of June 25, 06. These data are used for the air conditioner load control simulations (heater load control is performed through the similar approach).

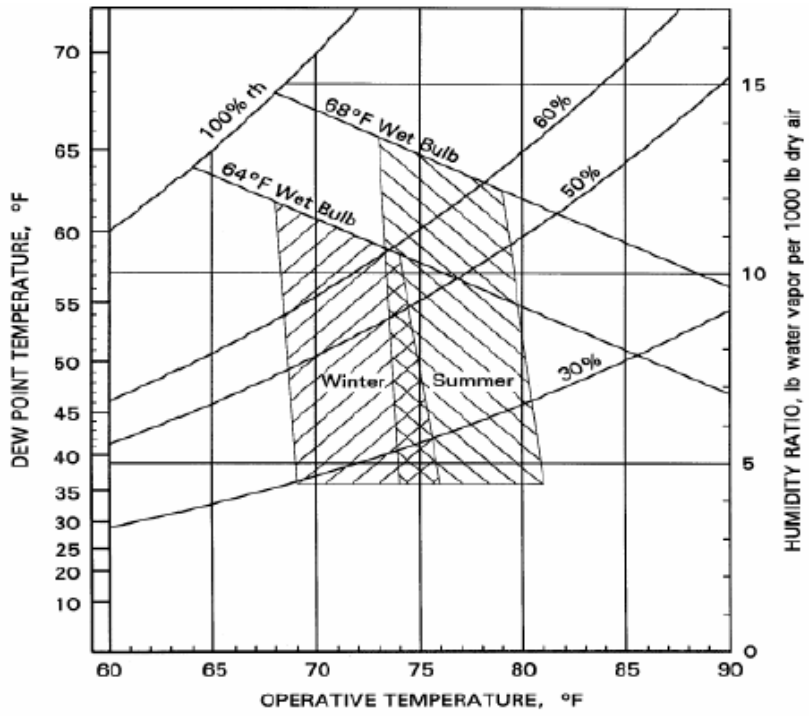


Figure 2.1 ASHRAE summer and winter comfort zones

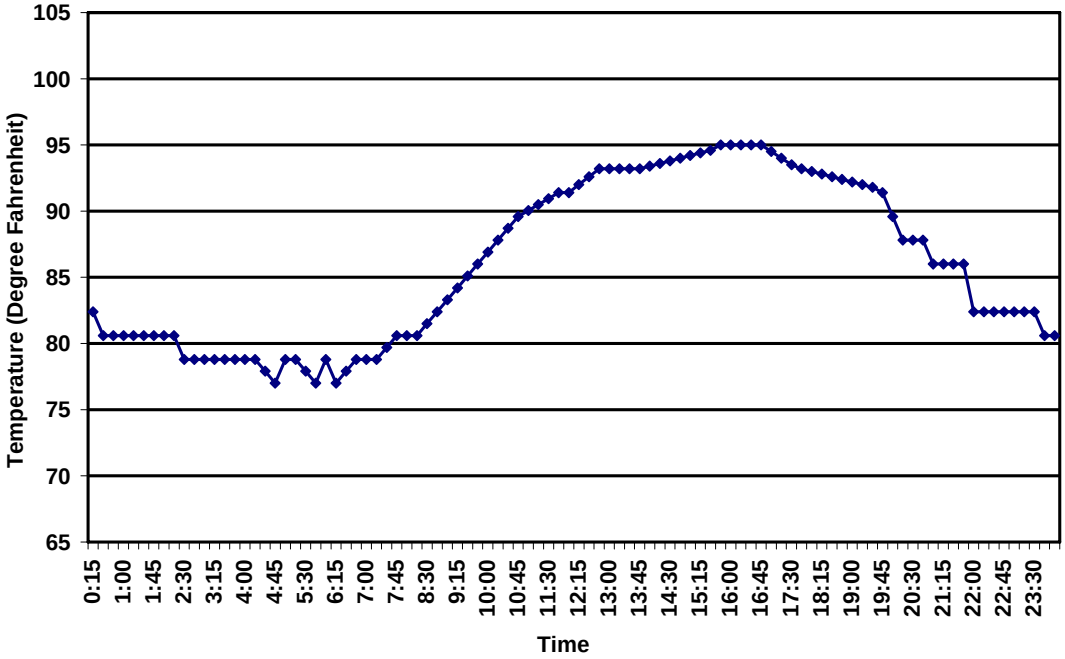


Figure 2.2 Dallas/Fort Worth area outdoor temperature (summer: June 25, 06)

2.1.2 Real Real-Time Pricing Data

Since the electricity real-time retail pricing data for end-use residential consumers in Texas are not available at the present time, the real-time price from www.ercot.com, the wholesale balancing energy service prices, are used as pseudo real-time consumer retail price data for the simulations. Fig. 2.3 shows the ERCOT's balance energy service market clearing prices of North area on a hot summer day of June 25, 06.

The objective function solves the minimization of the total cost, which is the summation of the product between the real-time price and energy consumed, subjecting to the consumers' comfort (temperature tolerances) and physical limit of air conditioner constraints. The optimization problem can be formulated as follows [4]:

$$\min C = \sum P_i q_i \tag{2.2}$$

subjects to

$T^{\min} \le T_i \le T^{\max}$ and

$0 \le q_i \le q^{\max}$

where $T^{\min} = T^{desired} - \text{allowed deviation}$

$T^{\max} = T^{desired} + \text{allowed deviation}$

q_i - energy (KWH) consumed for AC in hour i

P_i - price of electricity ($/MWH) in hour i

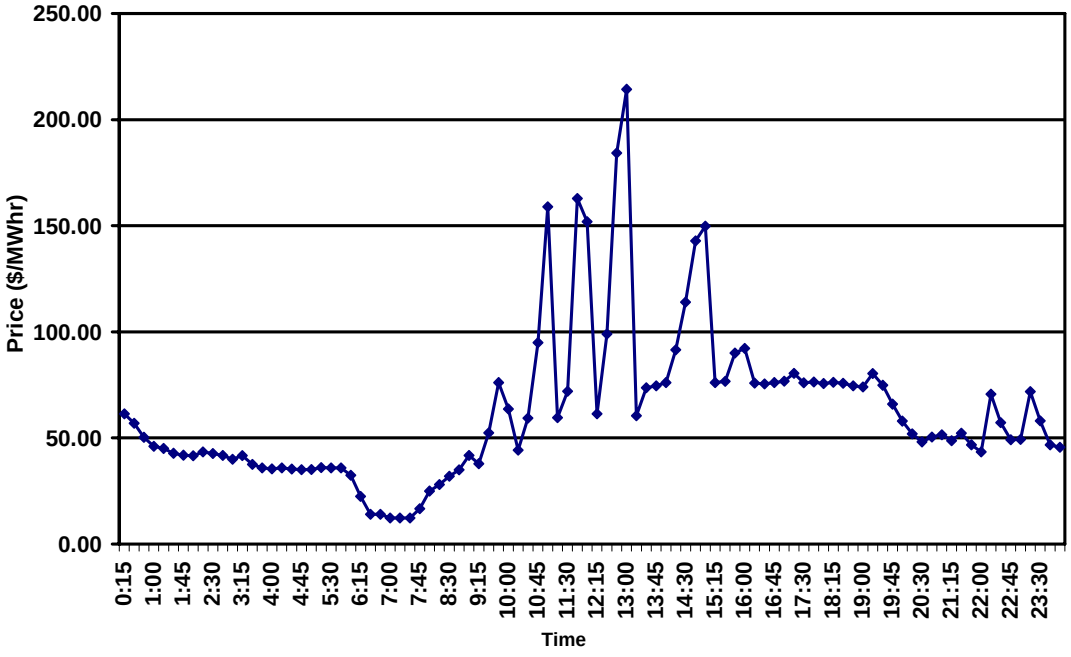


Figure 2.3 ERCOT Balance Energy Service Market Clearing Prices
- North area (summer: June 25, 06)

In the optimization problem, a four degree temperature band of thermostat setting is used to ensure the operative temperatures are always within the ASHRAE summer comfort zone regardless of the dew point temperature and humidity ratio (between 75 and 79 degree F, or plus and minus two degrees from the $T^{desired}$ of 77 degree F).

It is necessary to know all hourly data of prices and outdoor temperatures of the whole timeframe considered to solve the minimize problem of (2.2). In the real-time application, however, these data are not available. We cannot acquire all future data accurately to solve the minimize problem of the current time, and the forecast data are always imprecise (especially, the real-time electricity prices are determined by the market clearing prices from congestion management).

Thus, a strategy to control air conditioner load according to the available real-time data consumers perceive is proposed. The strategy is illustrated in Fig. 4. The air conditioner is operated the coolest possible (hence, near the lower boundary of the ASHRAE summer comfort zone; i.e., 75 degree), when the real-time price of electricity is lower than the predefined value (e.g., $10/MWH). On the other hand, the air conditioning load is operated the hottest (hence, near the upper boundary of the ASHRAE summer comfort zone; i.e., 79 degree), when the real-time price of electricity is higher than the predefined value (e.g., $40/MWH). The air conditioner is operated between these two boundaries depending on the real-time prices. This strategy is called the "Steps of Temperature Preferences," or the "Steps of Temperatures." Consumers

can easily adjust and balance the price and temperature settings according to their preferences and still maintain their comfort from the saving.

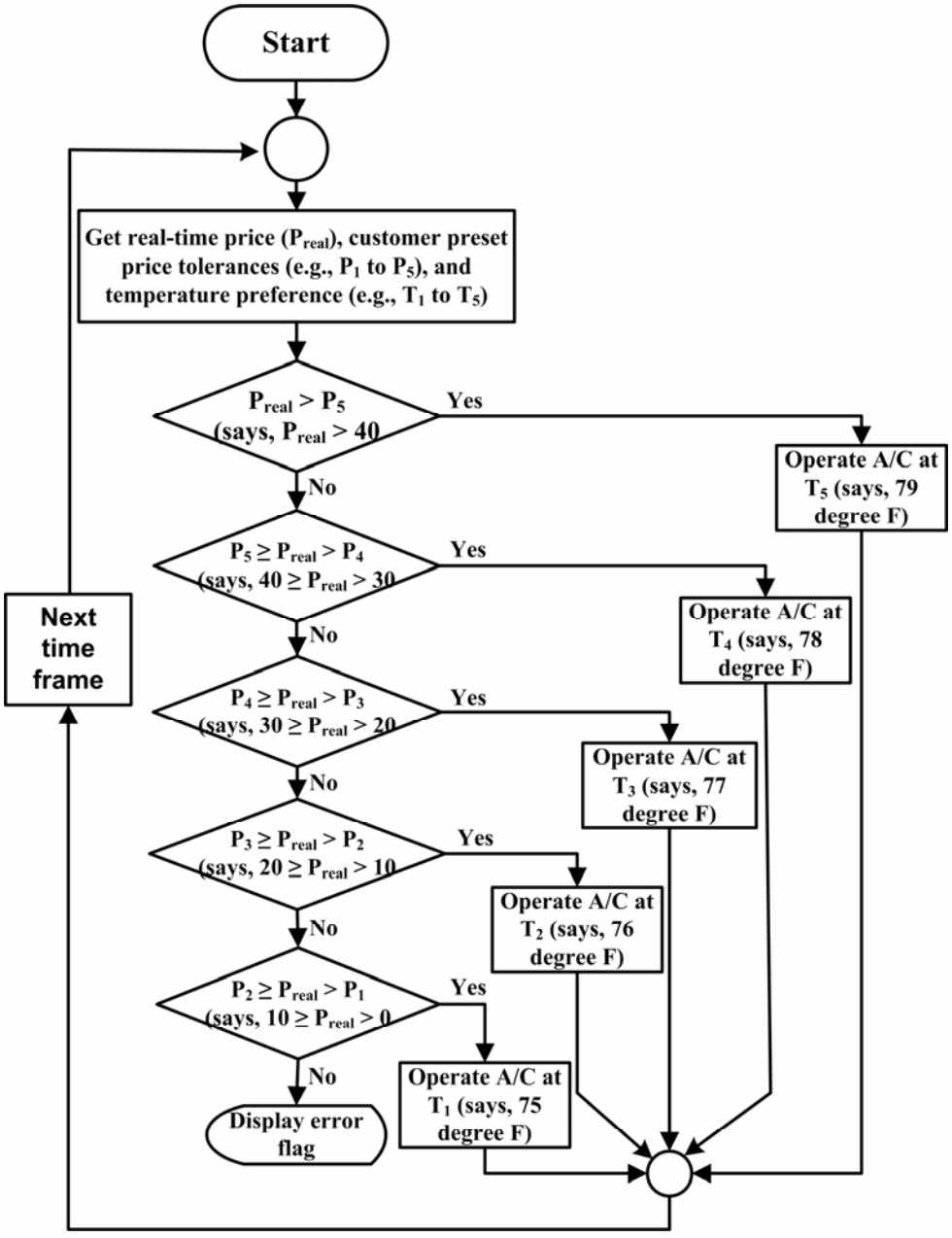


Figure 2.4 Proposed real-time air conditioning load control strategy

The proposed strategy is performed and compared its results with those from the optimization problem (using the MATLAB Optimization Toolbox [36]) and from the normal practice of a constant thermostat setting (no load control). The simulated indoor temperatures and the energy consumption are illustrated in Fig. 2.5 and 2.6 respectively. The total A/C consumption and total costs are shown in Table 2.1.

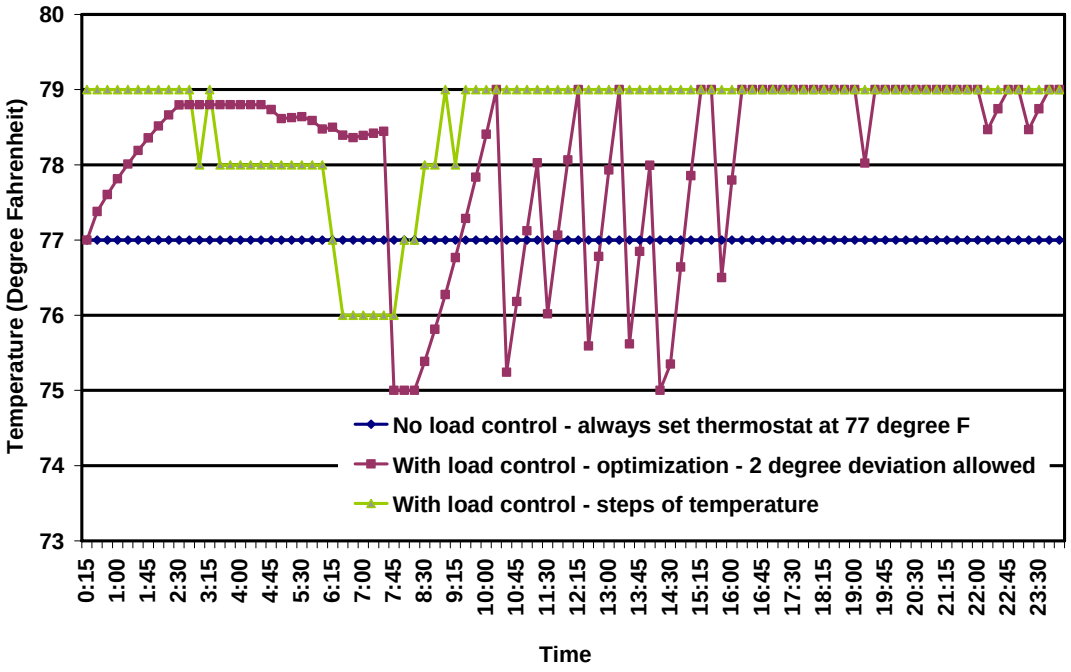


Figure 2.5 Simulated indoor temperature

The indoor temperatures with no load control remain the same at 77 degree regardless of real-time electricity prices. The temperatures with load controls vary between 75 and 79 degrees depending on the electricity prices and algorithms employed (e.g., optimization or steps of temperatures).

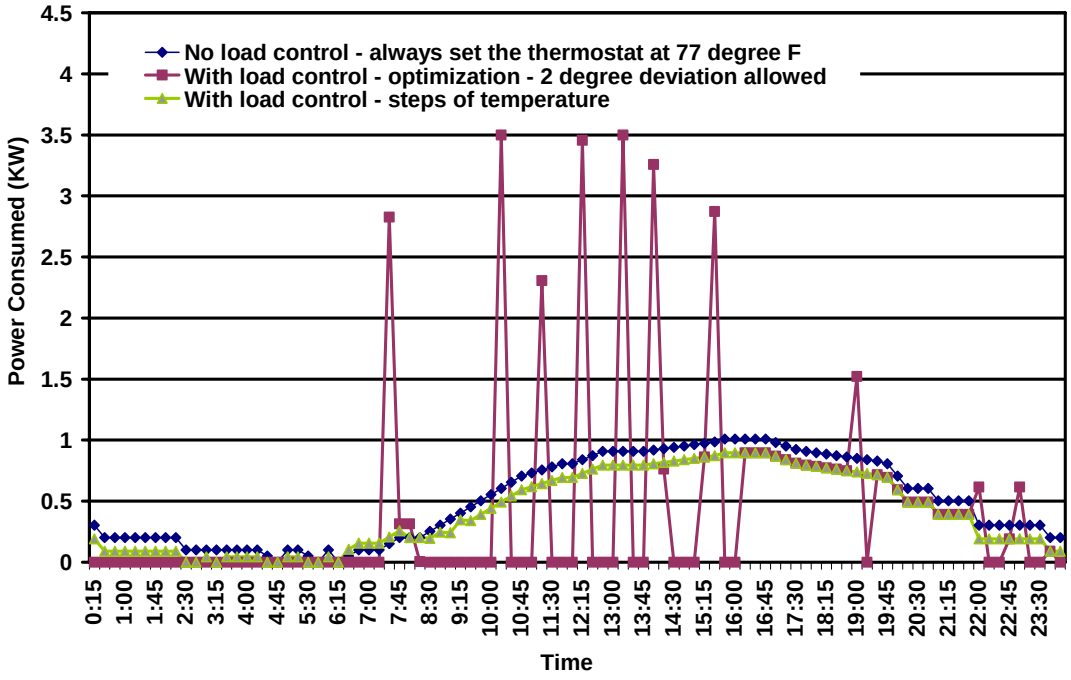


Figure 2.6 Simulated power consumption

Table 2.1 Results of AC load controls

Strategy	Consumption (KWH)	Total Cost ($)
No load control (always set at 77 degree F)	12.13	0.97
With Steps of Temperatures load control	10.03	0.82
Solving the minimized cost problem	10.28	0.63

The resulting total cost by using the Steps of Temperatures load control strategy is between the total costs by solving the minimized cost problem and by no load control. The total energy consumption, however, is even lower than that of the optimization problem. Note that, in the optimization, the target of the desired temperature is set to 77 degree F (the same as the temperature with no control strategy).

In summary, by controlling thermostat alone in one day, especially during the peak-price period, to allow only 2 degree deviation from the set point of 77 degree, consumers can potentially save cost of air conditioning load for \$0.97 - \$0.63 = \$0.34 (34.79%) by solving the minimized cost problem, and \$0.97 - \$0.82 = \$0.35 (15.98%) by the Steps of Temperature strategy. Consumers can save AC energy consumption of 12.13 - 10.28 = 1.85 KWH (15.23%) by the optimization strategy and 12.13 - 10.03 = 2.10 KWH (17.33%) by the Steps of Temperature strategy. They still can maintain their comfort from this saving. Please note that the resulting total costs are low because the wholesale (not retail) electricity prices are used in the simulations and the results are of 1 day only. When the real-time electricity pricing is in effect, utilities will add the profits to these wholesales prices before retailing electricity to their end-use consumers.

2.2 Water Heater Load Control

Water heater is another highly consumed domestic load. To control the water heater load, the similar strategy executed in the air conditioner/heater load control is applied. The results are compared with the normal practice of no load control, in which a household sets the operating point of the outlet hot water to a constant temperature.

The water heater load model previous discussed in [27] and [35] is employed as a basis in the simulations. The average daily hot water consumption (Gallons per Day or GPD) can be expressed as follows:

$$GPD = 24 + 0.016 \times CFA \tag{2.3}$$

where CFA or Conditioned Floor Area (ft^2) is assumed = 2,500 ft^2.

Hourly hot water consumption during the hour under consideration (Gallons per Hour or GPH) can be obtained by multiplying the GPD to the fraction of typical water heating schedules specified in the Fig. 2.7.

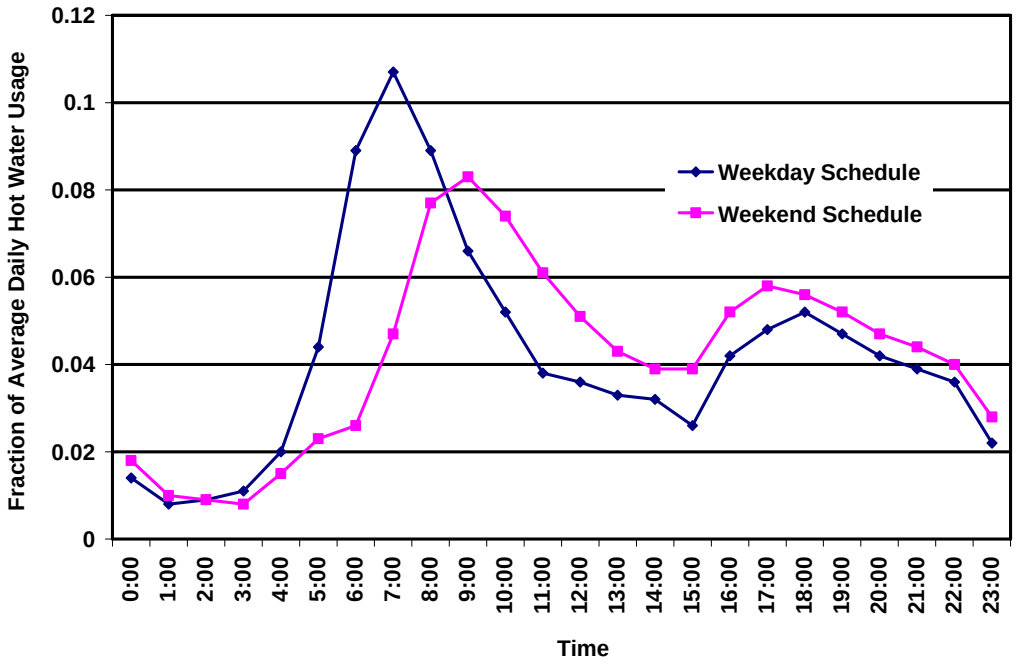


Figure 2.7 Typical hot water consumption schedules

Subsequently, the hourly load in watts can be obtained from (2.4).

$$\text{Hourly Water Heating Load (watts)} = \frac{8.3 \times GPH \times \Delta T}{3.412 \times EF} \tag{2.4}$$

where energy factor (EF) is between 0.7 and 0.95 (higher EF means more efficient water heater), and ΔT is the difference between inlet water and outlet hot water supply temperatures. In the simulation, the desired outlet hot water temperature is set to

120 degree F. The outlet hot water temperature of the no load control strategy is also set to a constant 120 degree F as a normal practice of a household.

The modified optimization problems for the water heater load control are to minimize the deviation of temperatures of the outlet hot water from the desired 120 degree (highest setting point), and, simultaneously, to minimize the total cost of energy consumed. These two objective functions are conflicting because the lower temperatures of the outlet hot water, the greater the savings (the lower of the total costs). These optimization problems are solved by the concepts of Multi-Objectives Optimization [39].

$$\min C = \sum P_i q_i \ , \quad \text{and}$$

$$\min \sum (T_{desired} - T^o) \tag{2.5}$$

subject to

$T^{\min} \le T^o \le T^{\max}$ ($T^{\max} = T^{desired}$ = 120 degree F)

The allowed temperature deviation is set, for example, = 10 degree F.

Thus, $T^{\min} = T^{desired}$ – allowed deviation = 120 – 10 = 110 degree F.

T^o - outlet hot water supply temperature (degree F)

T_i - inlet water temperature (degree F), assumed to be the real-time outdoor temperature at that hour

$\Delta T = T^o - T_i$

q_i - energy (KWH) consumed for AC in hour i

P_i - price of electricity ($/MWH) in hour i

As discussed earlier in the load control strategy of air conditioner/heater section, the optimization problem is not applicable during the real-time application because of the availability of the valid and accurate future data. Thus, the identical strategy in the air conditioner/heater load control is executed as shown in Fig. 2.8 to control the water heater and the same arbitrary and simple price step tolerances (i.e., 0, 10, 20, 30, and 40 in $/MWH) is used. Customers can always balance and adjust to other steps according to their preferences.

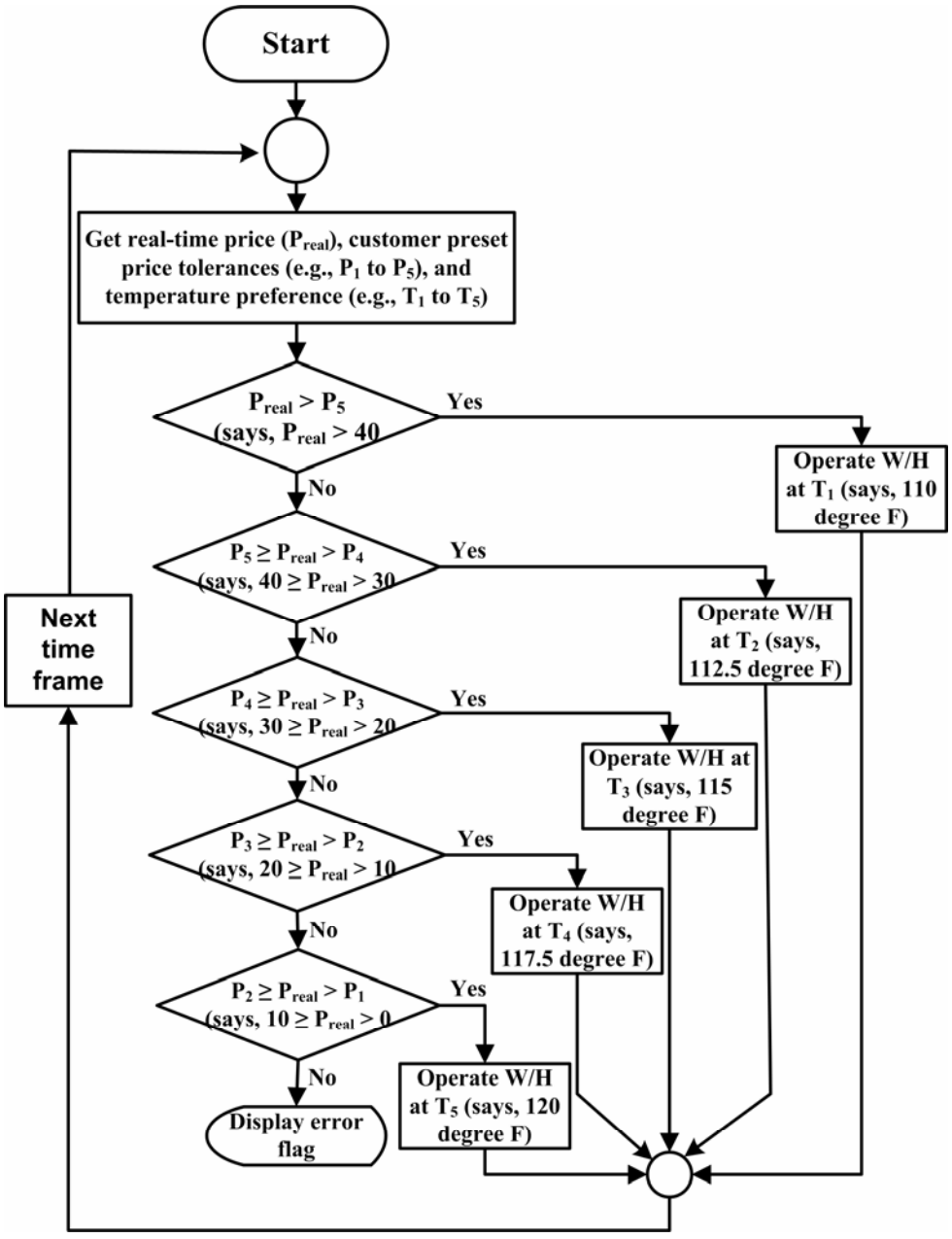


Figure 2.8 Proposed real-time water heater load control strategy

Unlike the air conditioner/heat load control strategy, the goal is to operate water heater the hottest allowable (max at desired 120 degree F) when the real-time price is the lowest (less than the predefined $10/MWH). When the price of electricity is higher

than the maximum price tolerance (higher than the predefined $40/MWH), the outlet hot water temperature is set to the lowest allowable (110 degree F). The water heater is operated between these two boundaries depending on the real-time electricity prices. The same ERCOT pseudo real-time prices and outdoor temperatures of summer June 25, 06 are used in the simulations. The results of hourly outlet hot water temperatures and corresponding energy consumed are illustrated in Fig. 2.9 and Fig. 2.10.

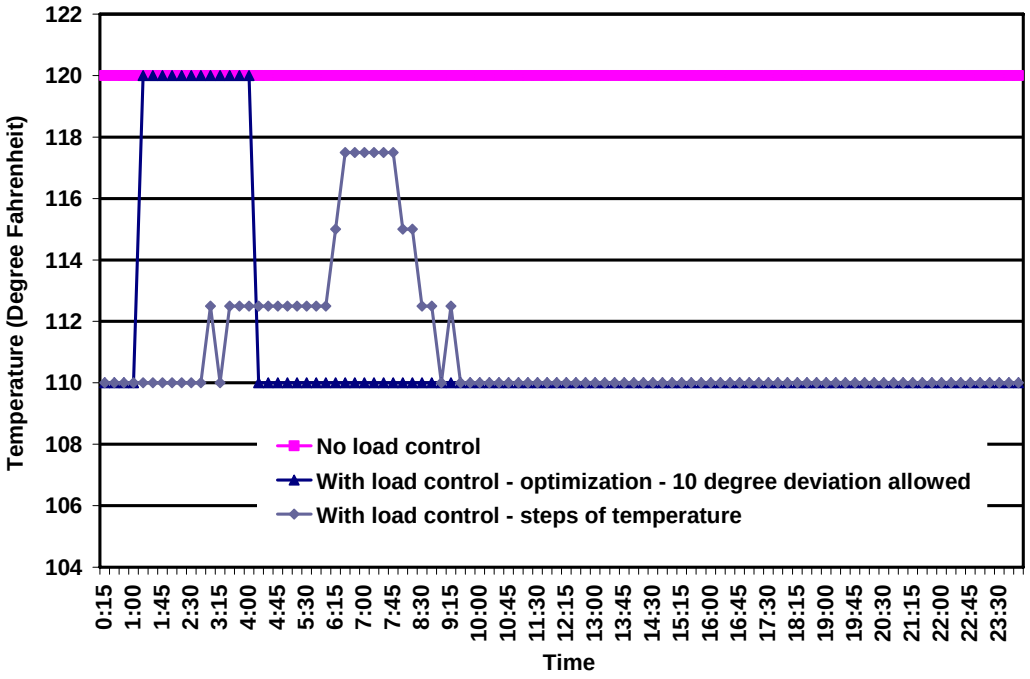


Figure 2.9 Simulated hourly outlet hot water temperature

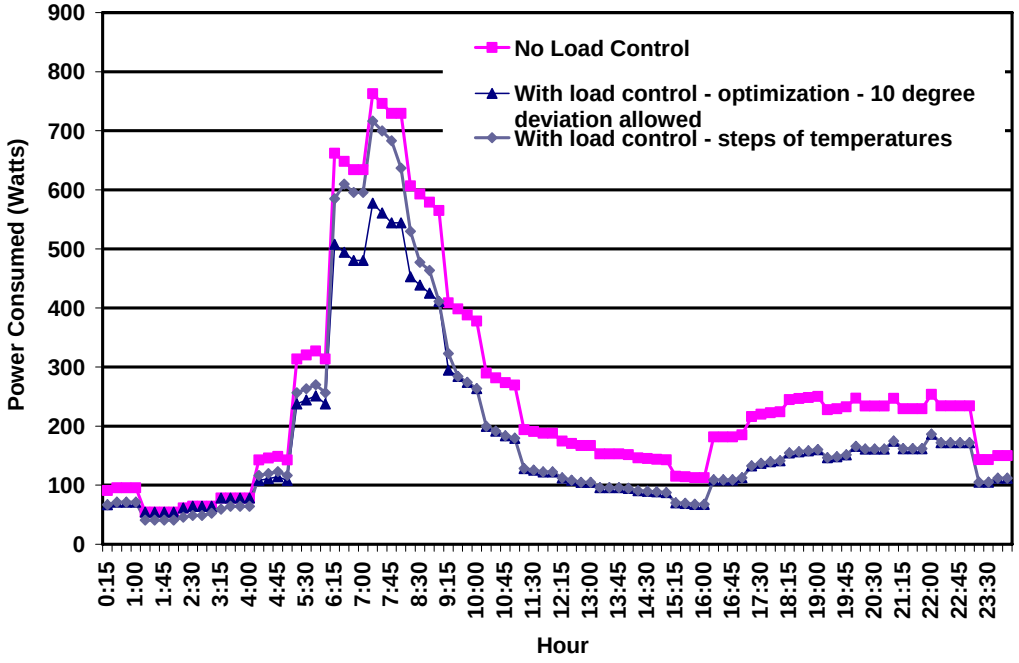


Figure 2.10 Simulated hourly water heating load

Table 2.2 Results of water heater load controls

Strategy	Consumption (KWH)	Total Cost ($)
No load control (always set at 120 degree F)	5.91	0.313
With Steps of Temperatures load control	4.49	0.218
Solving the minimized cost problem	4.22	0.215

In the case of optimization, the water heater operates at outlet hot water = 120 degree F only when both real-time prices and hourly water heating load are low (between 1 and 4 a.m.) and operates at 110 degree F in other timeframes. For the Steps of Temperature strategy, the water heater operates at temperatures different from 110

degree F (the lowest setting point) when the real-time electricity prices are low (between 3 and 8:30 a.m.)

To conclude, in one day, by controlling water heating load alone to allow 10 degree F deviation from normal operation of 120 degree F especially the during high electricity price period, customers can potentially save cost of $0.313 - $0.218 = $0.095 (31.2%) by solving the minimized cost problem, and $0.313 - $0.215 = $0.098 (30.4%) by the Steps of Temperature strategy. Consumers can save energy consumption of 5.91 - 4.22 = 1.69 KWH (28.52%) by solving the minimized cost problem and 5.91 - 4.49 = 1.42 KWH (24.08%) by the Steps of Temperature strategy. The strategy of the Steps of Temperature preferences in the water heater load control is effective because the total cost and power consumption are closed to the results from solving the optimization.

In both air conditioner/heater and water heater load controls, the performances of the savings depend on the real-time prices, real-time outdoor temperatures, and consumers' choices of price steps. In the example, the arbitrary and simple price steps (e.g., 0, 10, 20, 30, and 40 in $/MWH) are chosen as a demonstration. The savings, however, can be greater if the consumers are more experienced.

CHAPTER 3

PROPOSED LOAD CONTROL STRATEGY: PRICE NAMING

Chapter 2 proposes a consumer-centered load control strategy in real-time electricity pricing environment for the highest household electricity consumed loads according to [30] and [31] (i.e., air conditioner/heater and water heater). The strategy covers the re-schedulable usage loads mentioned earlier. Customers can adjust the output temperatures (i.e., air temperature for air conditioner/heater and outlet hot water for water heater) to take advantage of low real-time price periods for the load controls. In this chapter, on the other hand, another consumer-centered load control strategy for the re-schedulable usage and service loads such as cloth washer/dryer and dish washer is introduced. These loads can be deferred their usages to any other timeframes. The familiar concept of the on-line discount reservation in [40] is applied to the load control strategy. In other words, a residential customer can "name" his/her own electricity purchasing price for the load controller to send the control command to operate an appliance when the real-time electricity price drops below the desired price threshold. The approach can be applied to other re-schedulable usage and service loads as well. The objective is not to lower the overall consumption. Instead, the objective is to shift the appliances' load usages to subsequent cheaper electricity price timeframes, which in turn resulting in the cost savings to the consumers in the real-time pricing environment.

The strategy is tested using the pseudo real-time price data from ERCOT and the hourly load distribution probabilities of different home appliances [41]. The trade-offs between the cost-saving of each named price of each re-schedulable usage and service appliance comparing with its waiting time are investigated. The experiments are performed in MATLAB environment simulation.

3.1 Load Control and Simulations

3.1.1 Real-Time Pricing Data

The ERCOT wholesale real-time prices of North area are assumed to be the real-time prices residential customers perceive as in chapter 2. These sample pseudo hourly real-time prices of a hot summer between June 25 and 27, 06 (all are weekday) are shown in Fig. 3.1. The 3-day minimum price is $12.29/MWH (8AM of June 25), the 3-day maximum price is $206.10/MWH (11PM of June 27), and the 3-day average price is $49.88/MWH. If considered only one day of June 25, the minimum is $12.29/MWH (8AM), the maximum is $91.57/MWH (3PM), and the average is $54.73/MWH. Note that ERCOT provides real-time price every 15 minutes. The prices used in this chapter, for simplicity, are the prices at minutes 15 of the hours. Accordingly, the minimum and maximum prices of June 25 prices maybe different from those prices in chapter 2.

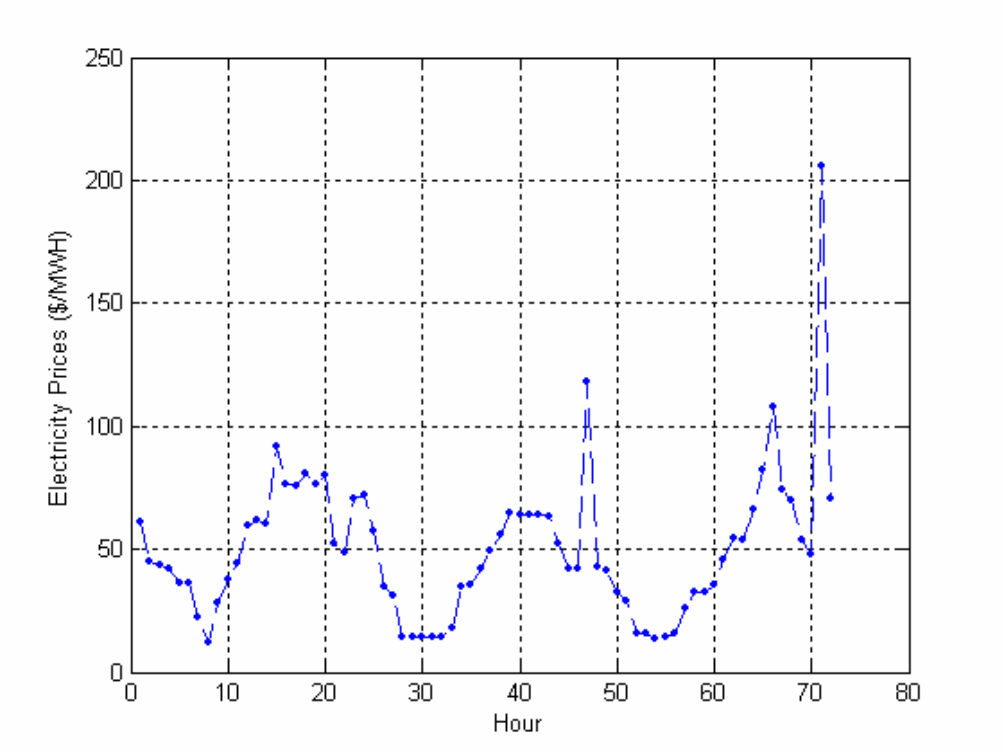


Figure 3.1 Pseudo residential hourly real-time prices (3 weekdays) from 12 AM of June 25, 06 to 12 AM of June 28, 06

3.1.2 Hourly Load Distribution Probability of Different Home Appliances

[41] provides the hourly load distribution probabilities of different home appliances. In the analysis of this dissertation, some samples of re-schedulable usage and service loads are tested, e.g., cloth washer/dryer and dish washer, which can be deferred their usages to other timeframes. These loads together account approximately 9.2% of a household consumption [31] and [32]. Their hourly load distribution probabilities are shown in Fig. 3.2 to 3.4 respectively.

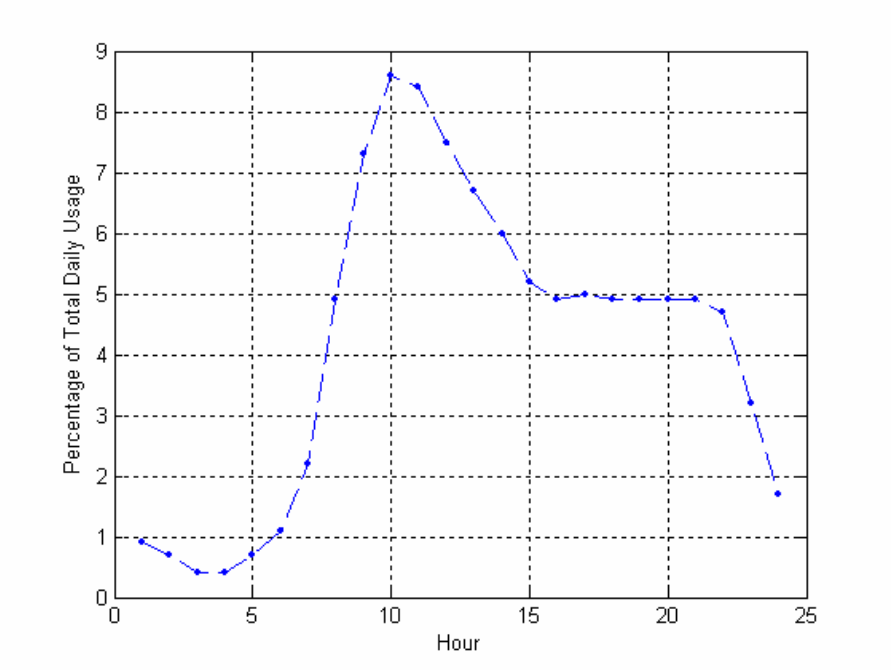


Figure 3.2 Hourly load distribution probability
of cloth washer (starting from 1:00 AM)

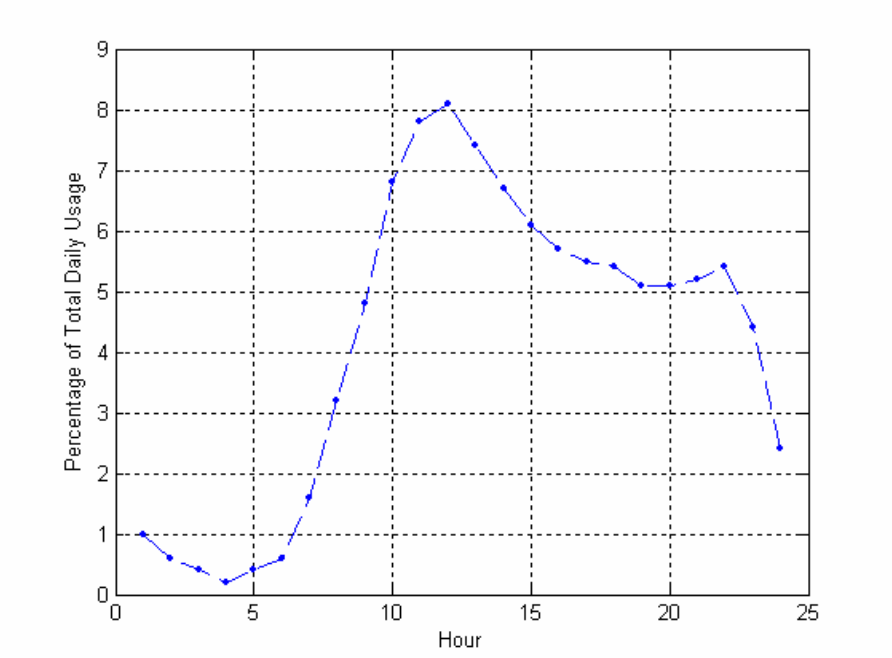


Figure 3.3 Hourly load distribution probability
of cloth dryer (starting from 1:00 AM)

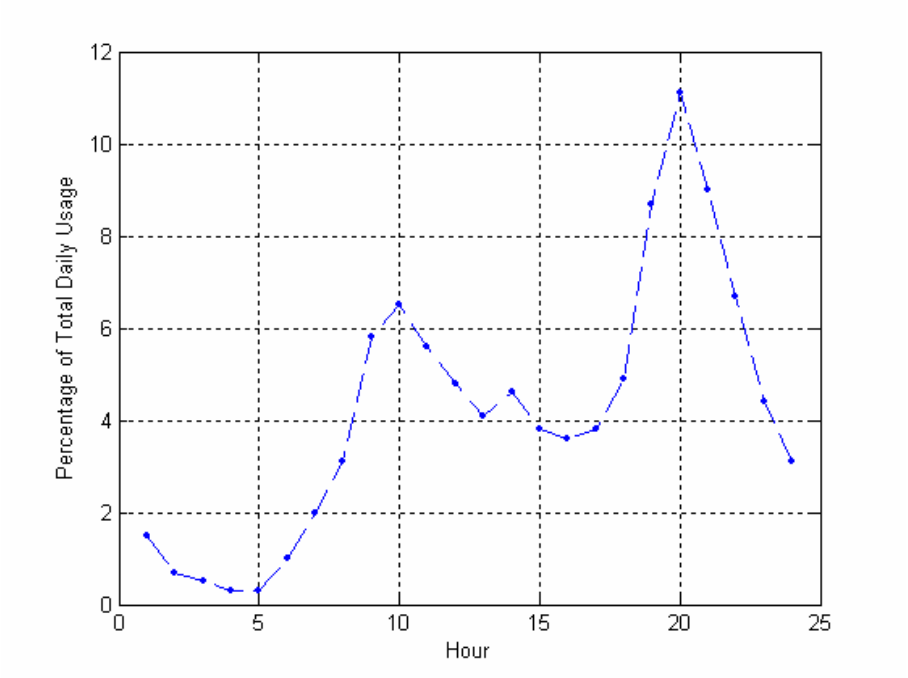


Figure 3.4 Hourly load distribution probability
of dish washer (starting from 1:00 AM)

For a typical household (1,000-3,000 ft^2), the yearly loads in KWH of the appliances can be approximately calculated as the functions of the number of bedrooms as follows [41]:

Cloth washer = 52.5 + 17.5 × number of bedrooms

Cloth dryer = 418 + 139 × number of bedrooms

Dish washer = 103 + 34.3 × number of bedrooms (3.1)

From (3.1), the daily loads of a typical household (3-bedrooms) in DFW areas assuming uniform daily load consumptions can be calculated as in (3.2):

Daily load of a household cloth washer = 0.288 KWH/day

Daily load of a household dryer = 2.288 KWH/day

Daily load of a household dish washer = 0.564 KWH/day (3.2)

Using (3.2) and Fig. 3.2 to 3.4, the aggregate hourly loads in MWH of typical 1,000 households can be approximated as in Fig. 3.5 to 3.7:

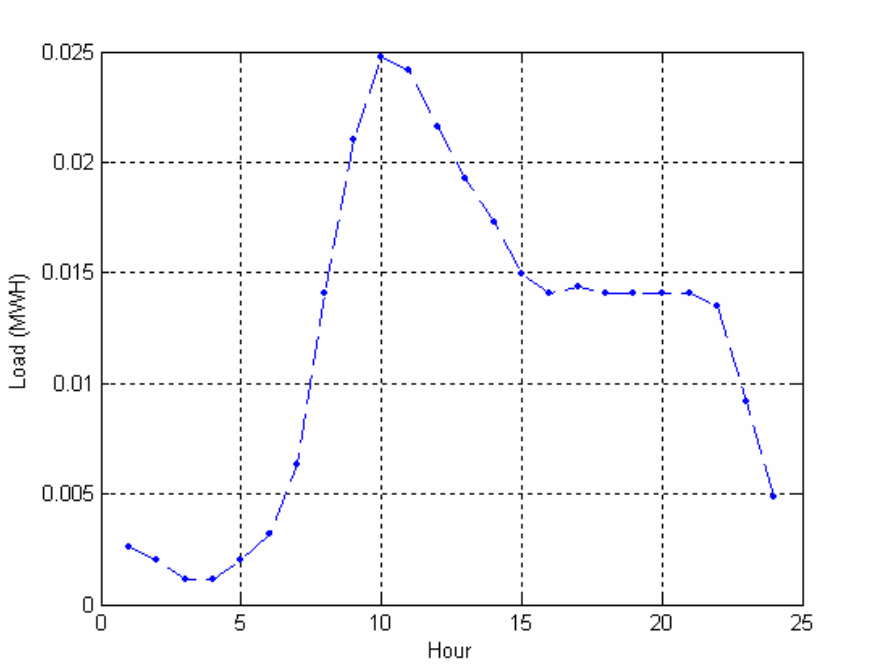


Figure 3.5 Approximate cloth washer hourly load profile
in MWH of typical 1,000 households

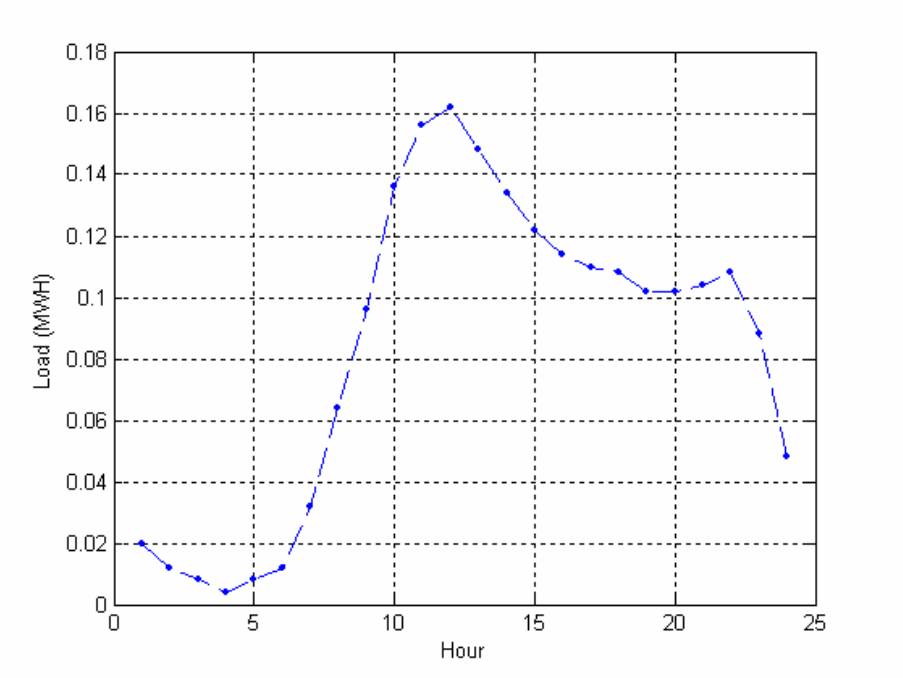


Figure 3.6 Approximate cloth dryer hourly load profile in MWH of typical 1,000 households

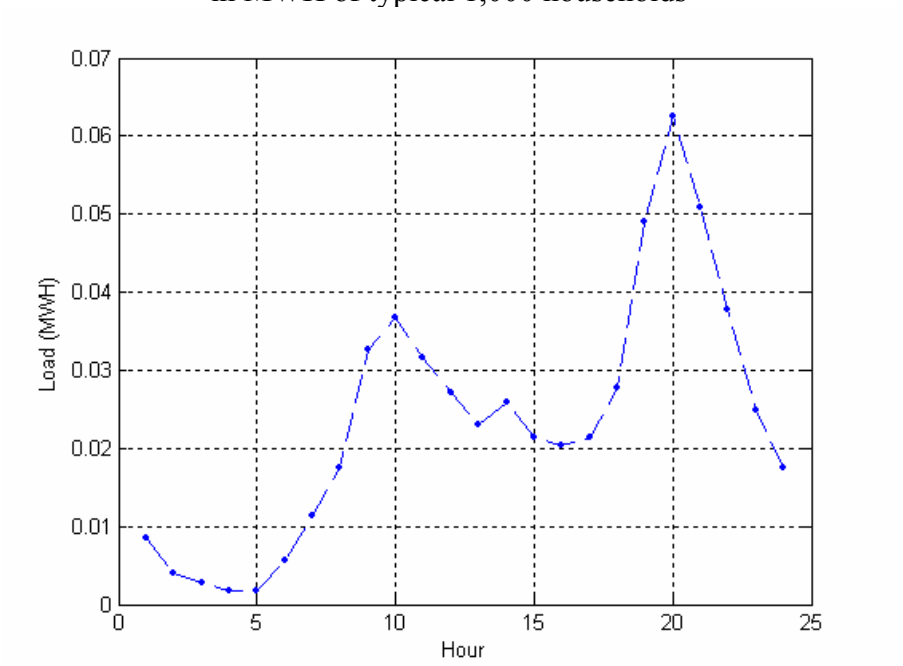


Figure 3.7 Approximate dish washer hourly load profile in MWH of typical 1,000 households

When accounting with the real-time prices of June 25, 06 from Fig. 3.1, the aggregate daily costs of 1,000 households can be calculated as in (3.3):

Aggregate daily cost for cloth washer = $12.196

Aggregate daily cost for cloth dryer = $136.453

Aggregate daily cost for dish washer = $33.552 (3.3)

(3.3) is the approximate aggregate consumption cost of each considered appliance with no load control and is the base case in the simulations of this chapter.

3.1.3 Price Naming Load Control Strategy

The concept of the familiar "price naming" is applied to the load control strategy. A customer can preset or "name" a price, which is acceptable (low enough) for him/her to start to operate an appliance. He/She, otherwise, prefers waiting and deferring the load consumption if the current real-time price does not satisfy this preset value.

The occurrences of the 3 weekdays (between June 25 and June 27, 06) that the real-time prices below customers' preset thresholds are tabulated as follows:

Table 3.1 Occurrences of 3 weekdays (June 25 to 27, 06) that real-time prices below customers' preset named price thresholds

Named Prices	Total Occurrences	Daily Average Occurrences
250	72	24.000
100	69	23.000
90	68	22.667
80	65	21.667
70	58	19.333
60	48	16.000
50	40	13.333
40	27	9.000
30	16	5.333
20	12	4.000
10	0	0.000

Table 3.1 illustrates the price naming chances or probabilities for a customer to attain his/her named prices. The lower the prices a customer names, the more the savings he/she can receive, but possibly the lower the chances he/she will attain those prices, and the longer the waiting times he/she need to defer the load consumptions. If a customer names the price too low (says, $10/MWH, which has the total occurrence = 0, or no occurrence), the appliance can not be operated at all timeframes during that 3-day period. On the other hand, if a customer names the price too high (says, $250/MWH, which has the daily average occurrences = 24, or always occurs), though he/she does not have to defer the operation of cloth washer/dryer or dish washer (no waiting time), he/she may not benefit from the cost savings of the price-naming demand response program.

To illustrate the strategy, first, the case of the aggregate loads when a group of all typical 1,000 household name the price for these 3 loads at the same $80/MWH is

simulated. The completed usage cycles of these loads are assumed to be one hour and the load consumption of each appliance type is assumed to be uniform. To demonstrate, the case for the named price of $80/MWH with the resulting load profiles and the waiting times is exhibited in Fig. 3.8 to 3.12.

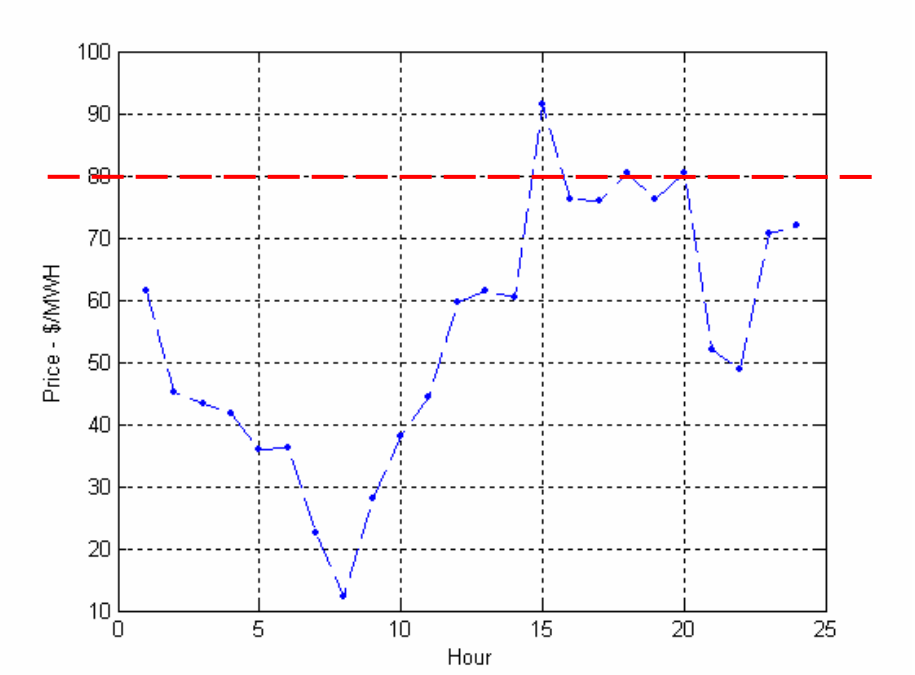


Figure 3.8 Real-time prices of June 25, 06
indicating the named price of $80/MWH

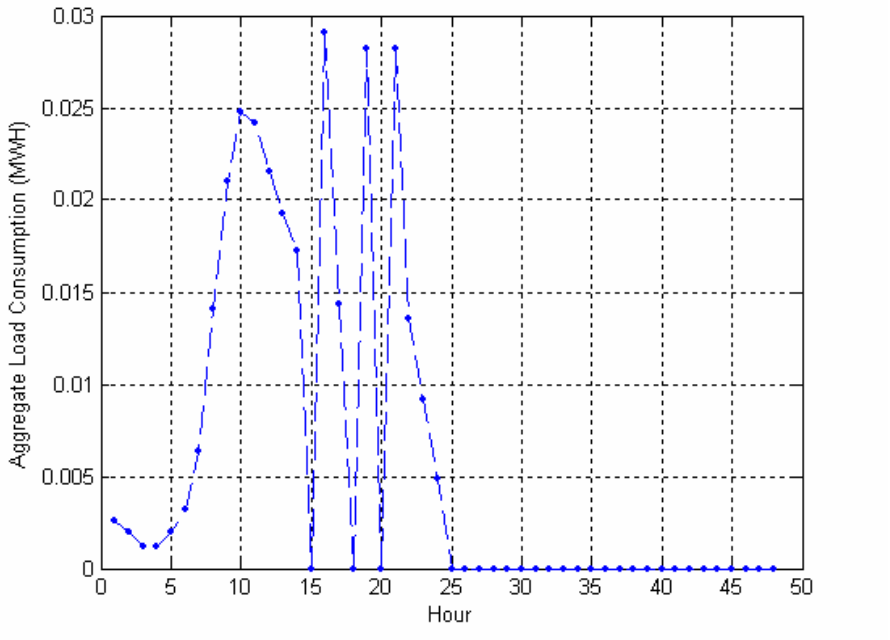


Figure 3.9 Simulation results of aggregate cloth washer hourly load profile, all customers name price at $80/MWH - only load of the first day (June 25, 06) is in consideration

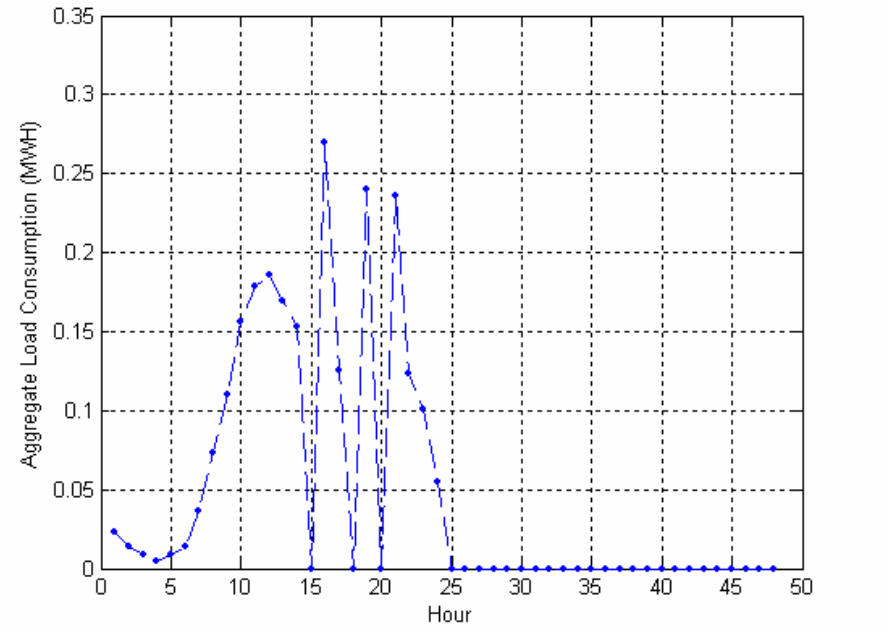


Figure 3.10 Simulation results of aggregate cloth dryer hourly load profile, all customers name price at $80/MWH - only load of the first day (June 25, 06) is in consideration

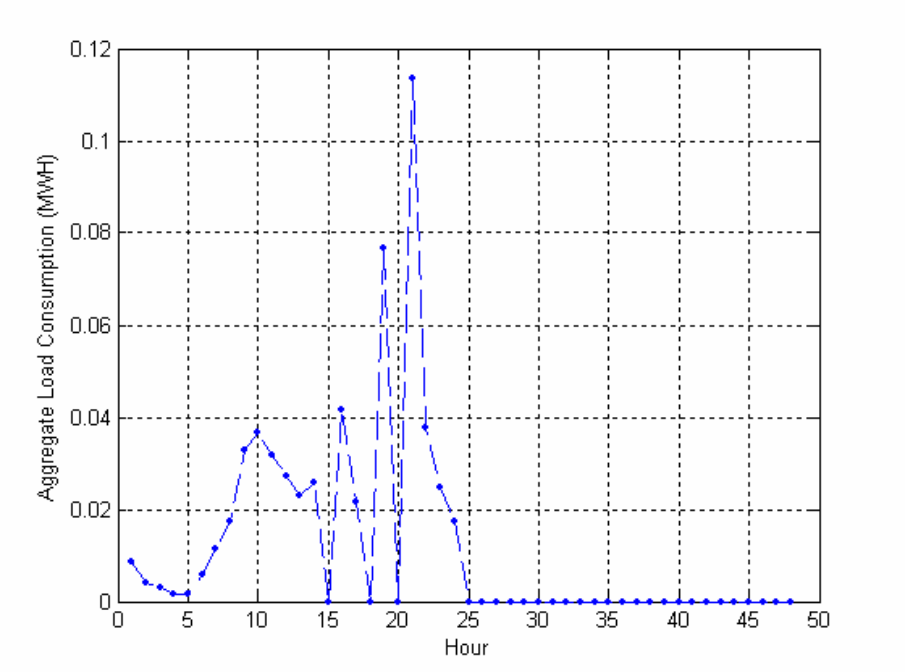


Figure 3.11 Simulation results of aggregate dish washer hourly load profile, all customers name price at $80/MWH - only load of the first day (June 25, 06) is in consideration

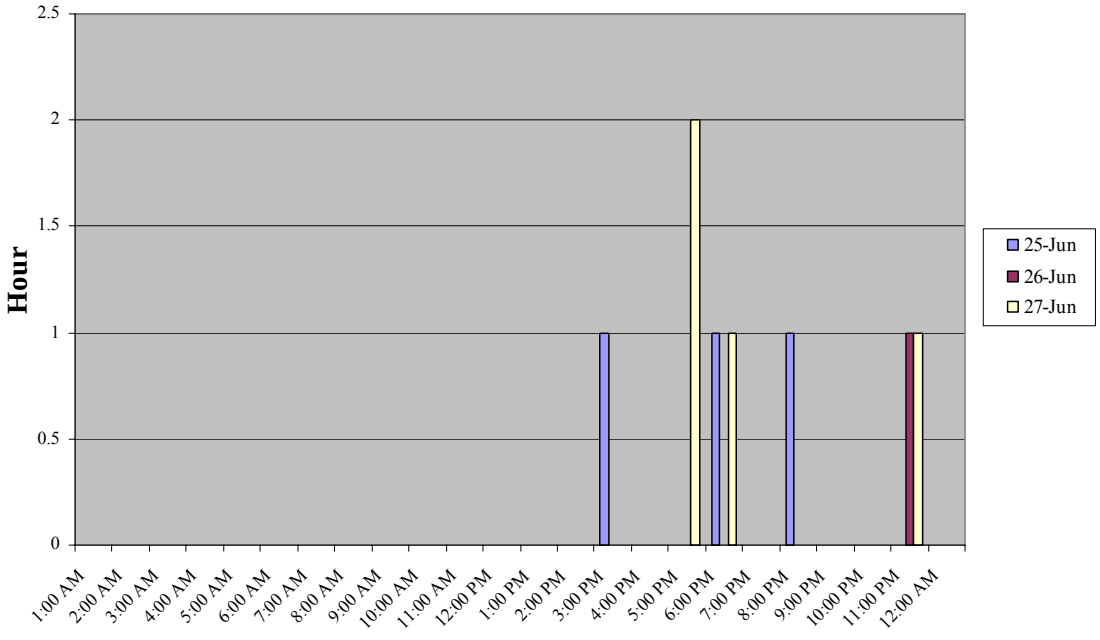


Figure 3.12 Waiting times for the named price of $80/MWH

From Fig. 3.8 to 3.12, the real-time electricity prices between midnight and 3PM of June 25, 06 were lower than the customers' named price of $80/MWH. The load controller can operate these appliances immediately with no waiting time during this period. At 3PM, however, the real-time electricity price was $91.57/MWH. The consumers who wish to start the appliances at that time choose to defer the loads to other period that have the real-time price lower than $80/MWH (at 4PM, the real-time price was $76.07/MWH). The waiting time, in this case, to defer the usage from 3PM to 4PM is one hour. This same approach is applied to all subsequent timeframes in that day (the real-time prices were also lower than $80/MWH at 6PM and 8PM with the waiting times of 1 hour as shown in Fig. 3.12). The waiting times are considered as the trade-offs comparing with the cost savings from participating in the price naming program.

Considered only one day of June 25, with all customers name the price at $80/MWH (high price naming scheme), the potential savings are 4.273% for cloth washer, 4.406% for cloth dryer, and 6.65% for dish washer comparing with the no load control case. Please note that if the consumers named price at a lower price, says, at $50/MWH as in Fig. 3.13, the potential savings will be 24.554% for cloth washer, 26.528% for cloth dryer, and 26.848% for dish washer but they may have longer waiting times. The consumers who named $50/MWH at 12PM have to wait for 10 hour before the real-time price dropped lower than $50/MWH at 10PM. The waiting times of the corresponding named prices for all appliances are the same because of the same proportions of the named prices in each case.

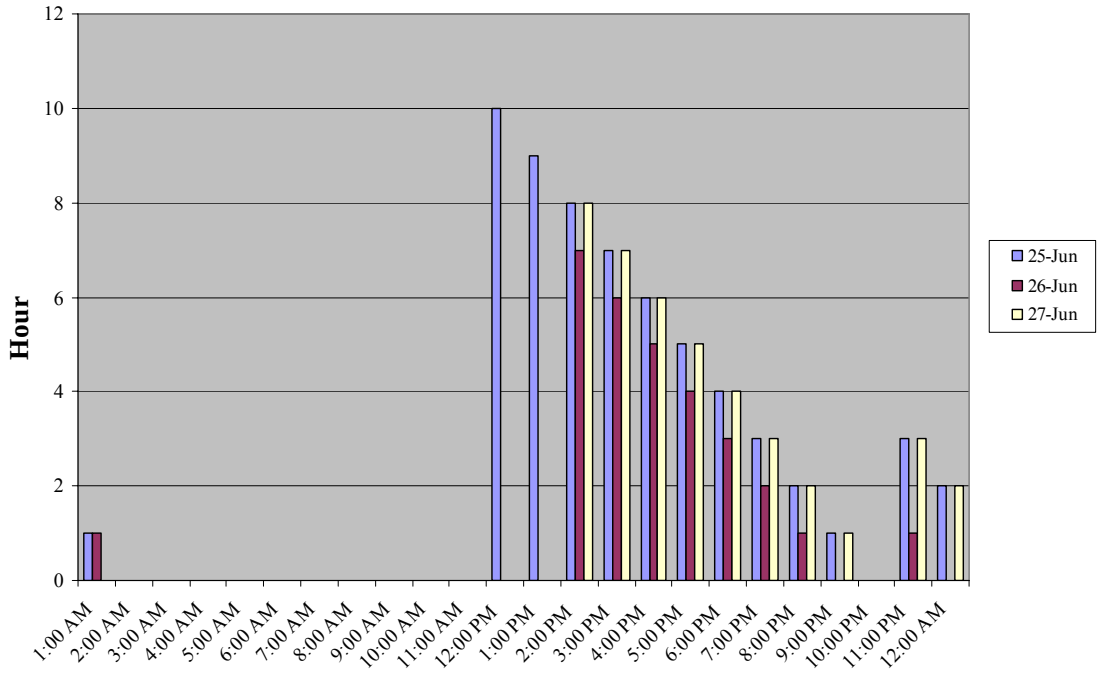


Figure 3.13 Waiting times for the named price of $50/MWH

Next, consumers are assumed to have different degrees of preferences or interests in participating in the price naming load control program. A household customer can also name the price differently for different appliances. To simplify the illustration, the percentages are distributed as shown in Table 3.2:

Table 3.2 Customers' participation percentages in
real-time price naming load control program

	Percentage of Participation
Not participate in load control	40
Named price = 20	10
Named price = 30	10
Named price = 40	10
Named price = 50	10
Named price = 60	10
Named price = 70	10
Total	100

The resulting MATLAB simulations of these load profiles are presented in Fig. 3.11 to 3.13. The cost savings are included in Table 3.3 to 3.5.

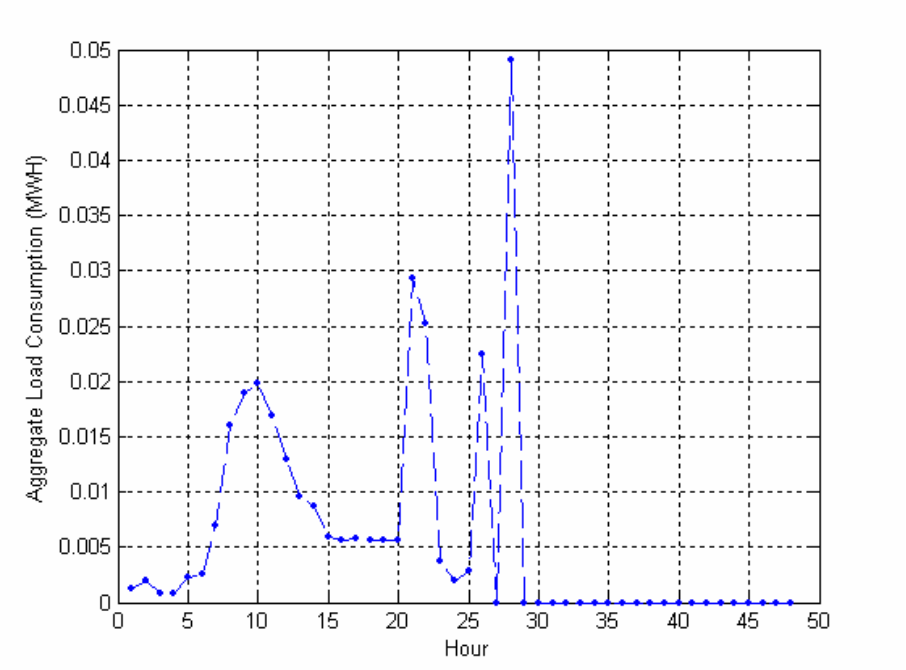


Figure 3.14 Simulation results of aggregate cloth washer hourly load profile,
only load of the first day (June 25, 06) is in consideration

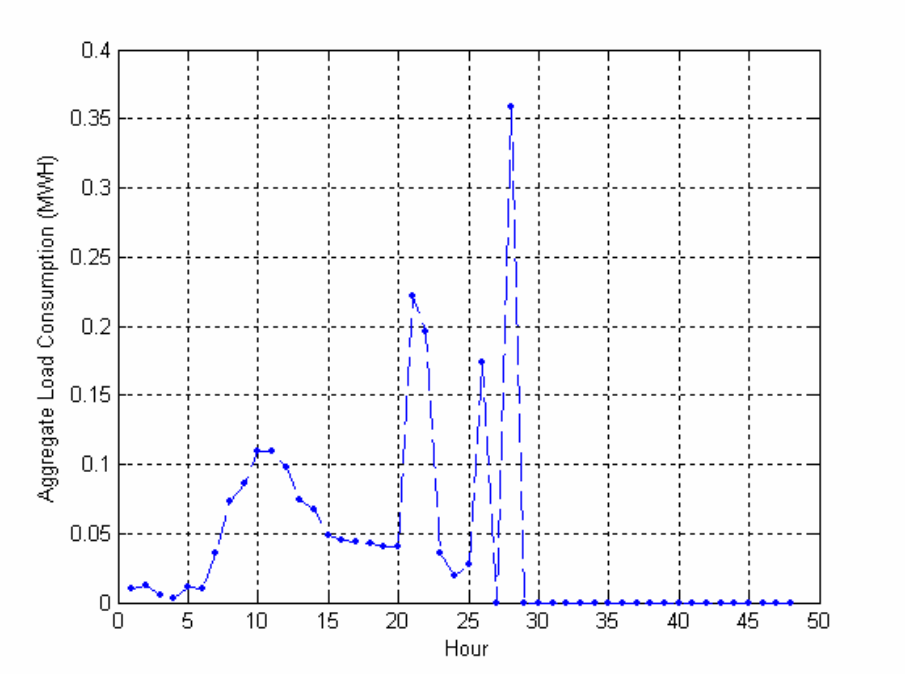


Figure 3.15 Simulation results of aggregate cloth dryer hourly load profile, only load of the first day (June 25, 06) is in consideration

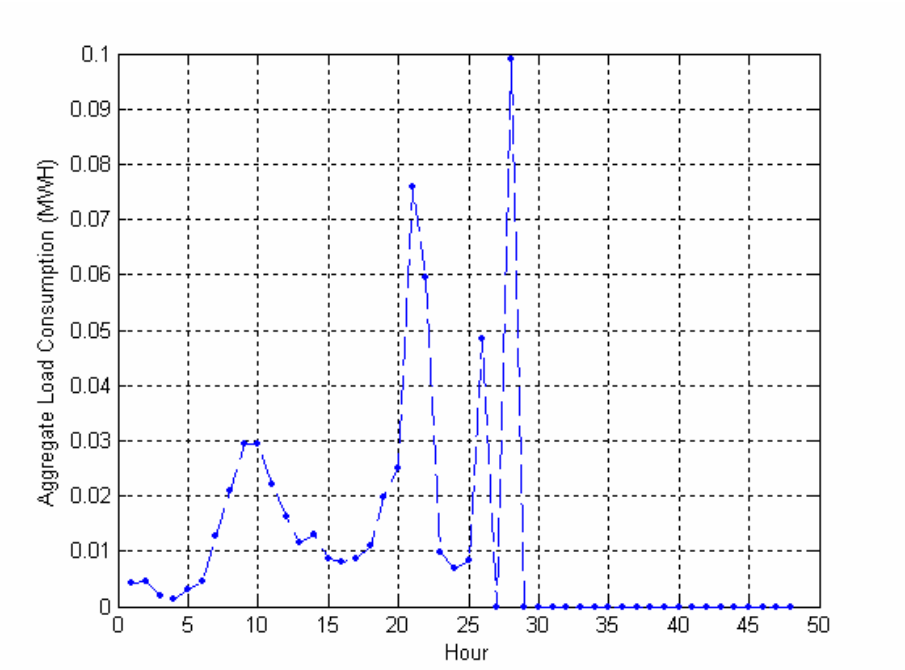


Figure 3.16 Simulation results of aggregate dish washer hourly load profile, only load of the first day (June 25, 06) is in consideration

In the simulations, the daily load consumptions of only the first day (June 25, 06) are in the consideration. The resulting load consumptions on the second day (June 26, 06) appeared in Fig. 3.11 to 3.13 are the shifted loads that otherwise consumed on June 25 had the electricity real-time prices met the customer's preset thresholds.

From the simulations results, some loads during the high prices on June 25, 06 are shifted to the off-peak price period (after midnight) of June 26, 06. Considered only one day, with all differences in customers' participation in the load control program as in Table 3.2, the potential savings are 24.841% for cloth washer, 25.868% for cloth dryer, and 26.025% for dish washer comparing with the no load control case.

Table 3.3 Aggregate cost savings of cloth washer load control (results of June 25, 06)

Named Prices ($/MWH)	Daily Total Cost ($)	Saving Comparing with No Load Control (%)
No Load Control	16.227	0.000
100	16.227	0.000
90	15.995	1.429
80	15.534	4.273
70	13.609	16.133
60	13.236	18.432
50	12.243	24.554
40	9.556	41.113
30	4.447	72.596
20	3.963	75.578
10	N/A	N/A
With Participation Differences from Table 3.2	12.196	24.841

Table 3.4 Aggregate cost savings of cloth dryer load control
(results of June 25, 06)

Named Prices ($/MWH)	Daily Total Cost ($)	Saving Comparing with No Load Control (%)
No Load Control	136.453	0.000
100	136.453	0.000
90	134.290	1.585
80	130.440	4.406
70	112.931	17.238
60	109.638	19.652
50	100.254	26.528
40	77.091	43.503
30	34.245	74.904
20	31.577	76.858
10	N/A	N/A
With Participation Differences from Table 3.2	119.294	25.868

Table 3.5 Aggregate cost savings of dish washer load control
(results of June 25, 06)

Named Prices ($/MWH)	Daily Total Cost ($)	Saving Comparing with No Load Control (%)
No Load Control	33.552	0.000
100	33.552	0.000
90	33.220	0.990
80	31.320	6.651
70	27.349	18.487
60	26.769	20.216
50	24.544	26.848
40	18.959	43.492
30	8.597	74.377
20	7.773	76.832
10	N/A	N/A
With Participation Differences from Table 3.2	33.552	26.025

To check the consistency of the results in Table 3.3 to 3.5, the same approach is performed on the next 2 weekdays (June 26 and June 27, 06). The average results of both weekdays are tabulated in Table 3.6 to 3.8 as follows:

Table 3.6 Aggregate cost savings of cloth washer load control
(average results of June 26 and 27, 06)

Named Prices ($/MWH)	Daily Total Cost ($)	Saving Comparing with No Load Control (%)
No Load Control	14.894	0.000
100	13.428	9.840
90	13.428	9.840
80	13.336	10.459
70	13.095	12.079
60	11.780	20.903
50	10.960	26.410
40	8.770	41.116
30	7.821	47.489
20	3.831	83.224
10	N/A	N/A
With Participation Differences from Table 3.2	11.583	22.227

Table 3.7 Aggregate cost savings of cloth dryer load control
(average results of June 26 and 27, 06)

Named Prices ($/MWH)	Daily Total Cost ($)	Saving Comparing with No Load Control (%)
No Load Control	128.458	0.000
100	113.207	11.873
90	113.207	11.873
80	112.400	12.501
70	110.058	14.324
60	97.951	23.749
50	90.755	29.350
40	71.430	44.394
30	63.421	50.629
20	30.250	76.451
10	N/A	N/A
With Participation Differences from Table 3.2	97.770	23.890

Table 3.8 Aggregate cost savings of dish washer load control
(average results of June 26 and 27, 06)

Named Prices ($/MWH)	Daily Total Cost ($)	Saving Comparing with No Load Control (%)
No Load Control	32.169	0.000
100	28.479	11.469
90	28.479	11.469
80	28.342	11.897
70	27.709	13.864
60	24.301	24.458
50	22.400	30.367
40	17.437	45.795
30	15.535	51.709
20	7.473	76.770
10	N/A	N/A
With Participation Differences from Table 3.2	24.353	24.296

The obtained results from Table 3.6 to 3.8 (averages of 2 weekdays between June 26 and 27, 06) are relatively consistent with the results from Table 3.3 to 3.5 (only June 25, 06). Accordingly, consumers can also roughly estimate the approximate amount of saving and the waiting time of each appliance corresponding to each price they name using this information.

CHAPTER 4

CUSTOMER'S USER INTERFACE

From the proposed load control strategies (i.e., Steps of Temperatures in chapter 2 and Price Naming in chapter 3), in this chapter, these strategies are implemented in the prototype using MATLAB programming for the customer's graphic user interface (GUI) and computational tasks. With a lowering price of a personal computer nowadays, a personal home computer is a familiar and common device in a household. MATLAB is an advanced, well-known, and affordable computational software that can easily be installed to a personal home computer. The source codes of the user interface are presented in the Appendix. The following sections will describe each mode of operation of the GUI.

4.1 Home Mode

The Home mode is initialized during the startup of the program. The onscreen GUI will display logo, version, and credits (Fig. 4.1). The user can change the mode of operation by selecting the menu on the top. The menu has the following customer's choices:

Home

X10_PLC_RF

Price_Naming_Assistant

Real_Time_Price

Real_Time_Outdoor_Temperature

Survellance_Camera

Figure 4.1 Print screen of the MATLAB customer's graphic user interface (GUI) in the Home mode

4.2 X10_PLC_RF Mode

When a user chooses the X10_PLC_RF from the top menu, the GUI will initiate the appliance load control mode named X10_PLC_RF as Fig. 4.2. The current time, current electricity price for the residential consumer in $/MWH (detailed in section 4.4),

and the current outdoor temperature in degree F (detailed in section 4.5) are also shown as information to the consumer for his/her load control decision makings.

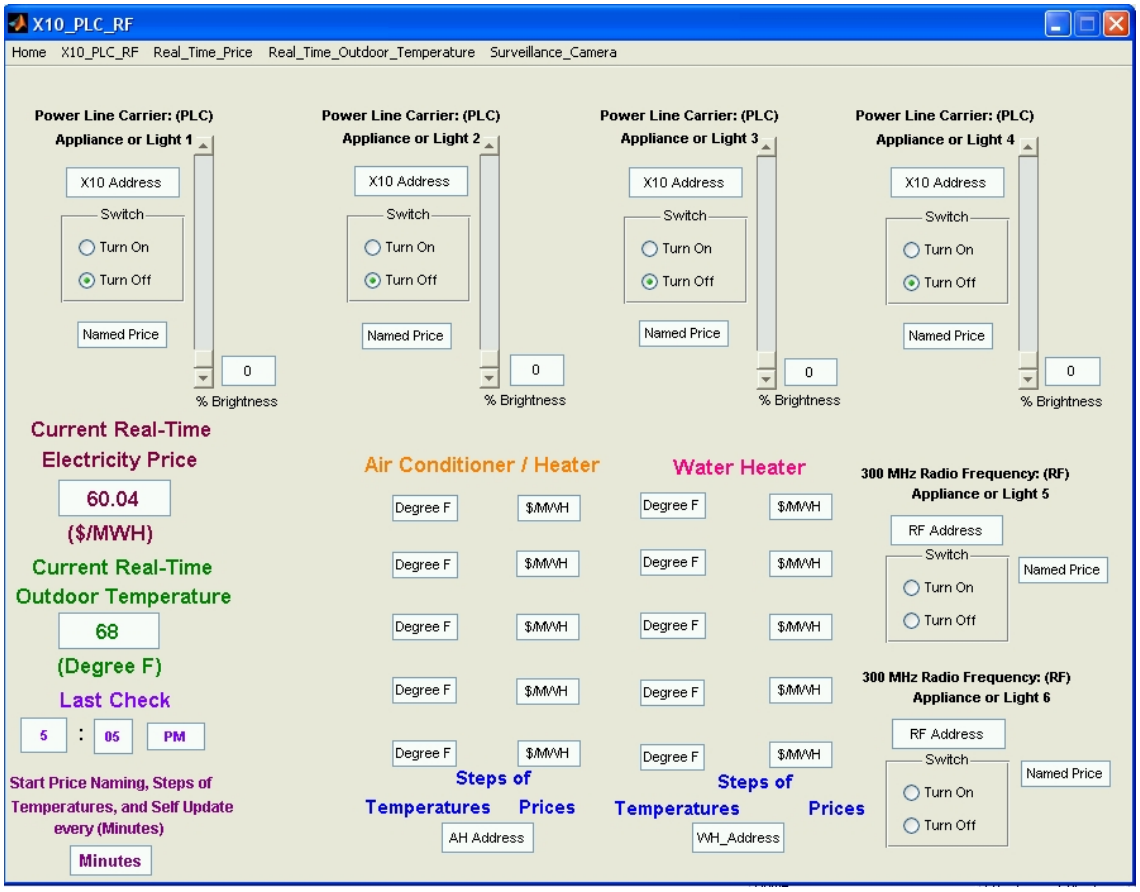


Figure 4.2 Print screen of the MATLAB customer's graphic user interface (GUI) in the X10_PLC_RF mode

The appliance load controls in the X10_PLC_RF employ 2 different communication protocols to send or receive load control commands between the computer load control and the loads: firstly, X10 power line carrier (PLC) to control appliances or lights number 1 to 4 and, secondly, 300 MHz radio frequency (RF) to control appliances or lights number 5 to 6.

X10 PLC protocol has long been widely used and has a major advantage in that it uses the existing home AC power line as a medium to transmit or receive data, thus, no additional costs to install new control signal wires. X10 devices are also inexpensive, simple to install, compatible among various manufacturers and able to be controlled up to 256 X10 addresses [42] and [43]. X10 protocol is one of the protocols in the prototype because of it is a well-known and long established home automation and is economical and sufficient for the demonstrations.

X10 signals are RF bursts which represent digital information synchronized to the zero crossing point of the AC power line: a binary 1 represented by a 1 millisecond burst of 120 kHz at the zero crossing point and a binary 0 by the absence of 120 kHz (Fig. 4.3) [44] and [45].

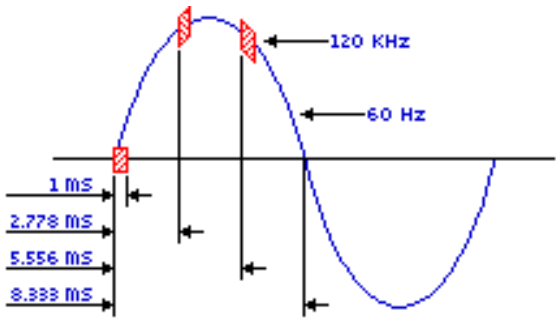


Figure 4.3 X10 signals synchronized to the zero crossing point
of the AC power line (showing timing relationship
for a three phase distribution system)

To transmit an X10 signal requires eleven cycles of the power line as depicted in Fig. 4.4. The first two cycles represent a Start Code (always binary 1110). The next four cycles represent a House Code from A (binary 0110) to P (binary 1100), corresponding to 16 combinations. The last five cycles represent either a Number Code

1 (binary 01100) to 16 (binary 11000), or a Function Code, for example, On (binary 00101) Off (binary 00101) as in Fig 4.5. This complete block, (Start Code, House Code, Key Code) should always be transmitted in groups of 2 as in Fig 4.6. With the combinations between the House Code A to P and the Number Code from 1 to 16, the X10 protocol can control up to 256 X10 devices (X10 addresses from A1 to P16).

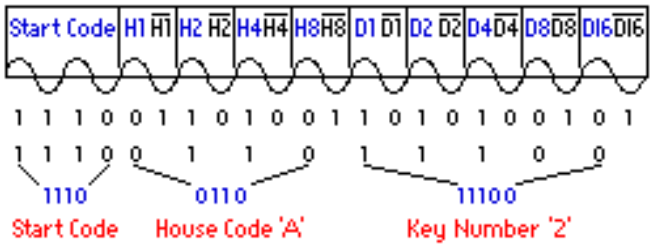


Figure 4.4 An X10 code encompasses eleven cycles of the power line

HOUSE CODES					KEY CODES					
	H1	H2	H4	H8		D1	D2	D4	D8	D16
A	0	1	1	0	1	0	1	1	0	0
B	1	1	1	0	2	1	1	1	0	0
C	0	0	1	0	3	0	0	1	0	0
D	1	0	1	0	4	1	0	1	0	0
E	0	0	0	1	5	0	0	0	1	0
F	1	0	0	1	6	1	0	0	1	0
G	0	1	0	1	7	0	1	0	1	0
H	1	1	0	1	8	1	1	0	1	0
I	0	1	1	1	9	0	1	1	1	0
J	1	1	1	1	10	1	1	1	1	0
K	0	0	1	1	11	0	0	1	1	0
L	1	0	1	1	12	1	0	1	1	0
M	0	0	0	0	13	0	0	0	0	0
N	1	0	0	0	14	1	0	0	0	0
O	0	1	0	0	15	0	1	0	0	0
P	1	1	0	0	16	1	1	0	0	0
					All Units Off	0	0	0	0	1
					All Lights On	0	0	0	1	1
					On	0	0	1	0	1
					Off	0	0	1	1	1
					Dim	0	1	0	0	1
					Bright	0	1	0	1	1
					All Lights Off	0	1	1	0	1
					Extended Code	0	1	1	1	1
					Hail Request	1	0	0	0	1
					Hail Acknowledge	1	0	0	1	1
					Pre-Set Dim	1	0	1	X	1
					Extended Data (analog)	1	1	0	0	1
					Status=on	1	1	0	1	1
					Status=off	1	1	1	0	1
					Status Request	1	1	1	1	1

Figure 4.5 Binary codes to be transmitted for each House Code and Key Code

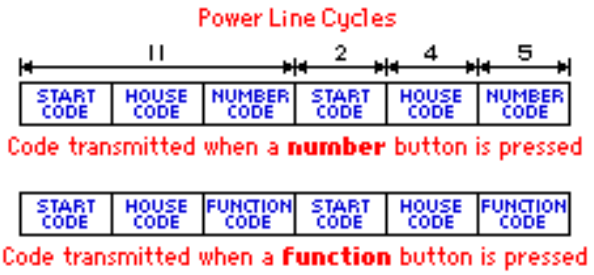


Figure 4.6 Complete blocks (Start Code, House Code, Key Code)
transmitted in groups of 2

Although some researchers provide the idea of X10 load controller using MATLAB [27], their MATLAB implementations are only for conceptual demonstration

and not detailed. In this dissertation's prototype however, the actual execution of MATLAB X10 controller is explained. X10 device model CM15A 2-way USB computer interface [46] is used to generate the X10 PLC signals from the MATLAB programmed load control commands. MATLAB interfaces with other external software applications through Component Object Model (COM). Fundamentally in Fig. 4.7, MATLAB can be configured as a computational server controlled by an external client COM component application program. On the other hand, when MATLAB controls another COM component, MATLAB is the client [36].

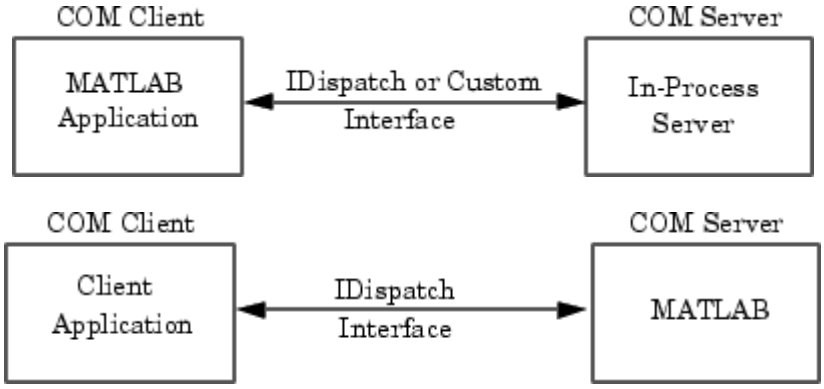


Figure 4.7 MATLAB Client/Server configurations [36]

The MATLAB command *h = actxserver('X10.ActiveHome.1')* is used to create an in-process COM server, and returns COM object, *h*, representing the server's default interface. The *X10.ActiveHome.1* is a programmatic identifier (progid) of the X10 device model CM15A 2-way USB computer interface component to instantiate in the server (see the Appendix for the source codes).

An application component that has a user interface, or an ActiveX control, is able to respond to actions taken by a user. The prototype interfaces with ActiveHome

Pro CM15A device through ActiveX scripting object. The ActiveHome scripting object has a *SendAction* function to transmit the power line carrier (PLC) command, *sendplc*, and to transmit the 300 MHz radio frequency (RF) command, *sendrf* [47]. The appliances have to be connected to the PLC protocol lamp module (e.g., model LM465 [48]) for incandescent light loads, PLC protocol heavy load appliance module (e.g., model AM486 [49]) for up to 15A appliance loads, or RF transceiver module (e.g., model TM751 [50]) for 300MHz RF protocol. In case where the PLC signals may be prone to noises [42], some PLC modules have the Automatic Gain Control (AGC) capability to filter out power line noises (e.g., model PAM02 [51]). In the prototype, the RF protocol appliance control is also employed as an alternate. To send a MATLAB complete appliance control command, an X10 address and additional parameters (all specified as string values) are also required to be included and correctly formatted (i.e., spaced and quoted). The field data structure *handles* is used to store large numbers of variables (e.g., X10 addresses, temperature set points, named prices). The command, for example:

handles.X10_address1=get(handles.X10_Address_Text1,'String')

is to create a variable *handles.X10_address1* from the user input X10 address typed in the fill-in box from the GUI. For demonstration at this instant, the fixed X10 addresses are shown as follows:

- *SendAction(h, 'sendplc', 'e11 on')* is to send the appliance control command through PLC protocol to turn on the appliance connected to the X10 device addressed E11.

- *SendAction(h, 'sendplc', 'a3 dim 80')* is to send the appliance control command through PLC protocol to dim the appliance connected to the X10 device addressed A3 to 80 percentage of its full brightness.
- *SendAction(h, 'sendrf', 'e off')* is to send the appliance control command through 300 MHz RF protocol to turn off the appliance connected to the X10 device addressed E.

The *SendAction* function can also be used to send the *queryplc* parameter to track the status of the modules in the system, for example:

- *SendAction(h, 'queryplc', 'b12 on'])* is to send the appliance control command through PLC to track the status of the appliance connected to the X10 device addressed B12. The returned parameter will be 1 if the appliance is on, 0 if the appliance is off, and -1 if the status is unknown.
- *SendAction(h, 'queryplc', 'g4 dim'])* is to send the appliance control command through PLC to track the brightness status of the appliance connected to the X10 device addressed G4. The returned parameter will be the dim level as a percentage from the full brightness, and -1 if the status is unknown.

When starting in the X10_PLC_RF mode, a customer can turn on or off X10 addressed appliances from the GUI similar to turn on or off normal physical switches. A consumer can also dim incandescent lights to percentages from the full brightness using either the sliders or filling in the desired percentages of brightness in the dim level boxes.

If a consumer prefers delaying the operations of appliances to take advantage of the lower electricity prices using the pricing naming load control strategy discussed in chapter 3, the consumer can name the prices in the fill-in boxes. The self-update time (in minutes) in refreshing the price data on the GUI is required to be set before the price naming process can be started (the MATLAB timer object is created using the command *timer* function).

To control X10 addressed air conditioner/heater (AH) and water heater (WH), a consumer can apply the Steps of Temperatures load control strategy discussed in chapter 2 to set the desired steps of temperatures corresponding to their steps of prices. The fill-in box for self-update time (in minutes) in refreshing the electricity price and weather data on the GUI has to be set before these AH and WH load controls can start. The controlling of AH and WH loads in X10_PLC_RF mode utilizes the PLC protocol to transmit the control signals.

The prototype uses TXB16 X10 thermostat [52] for home automation AH and WH load controls. When the TXB16 X10 thermostat receives the X10 PLC control commands from the MATLAB GUI, it can set the air conditioner/heater or water heater to the desired temperatures set points. The TXB16 user manual shows the X10 command table for the control protocol excerpted as follows in Table 4.1:

Table 4.1 Excerpted TXB16 X10 command protocol

Dim Level %	Preset Level	Send Set point			
		Unit 1	Unit 2	Unit 3	Unit 4
0	1	4	36	68	100
3	2	5	37	69	101
6	3	6	38	70	102
10	4	7	39	71	103
13	5	8	40	72	104
16	6	9	41	73	105
19	7	10	42	74	106
23	8	11	43	75	107
26	9	12	44	76	108
29	10	13	45	77	109
32	11	14	46	78	110
35	12	15	47	79	111
39	13	16	48	80	112
42	14	17	49	81	113
45	15	18	50	82	114
48	16	19	51	83	115
52	17	20	52	84	116
55	18	21	53	85	117
58	19	22	54	86	118
61	20	23	55	87	119
65	21	24	56	88	120
68	22	25	57	89	121
71	23	26	58	90	122
74	24	27	59	91	123
77	25	28	60	92	124
81	26	29	61	93	125
84	27	30	62	94	126
87	28	31	63	95	127
90	29	32	64	96	N/A
94	30	33	65	97	N/A
97	31	34	66	98	N/A
100	32	35	67	99	N/A

For example, to set the X10 addressed air conditioner load of House Code A to 90 degree F set point, the MATLAB GUI has to transmit the following command to the TXB16 thermostat:

SendAction(h, 'sendplc', 'a3 dim 71')

The thermostat, accordingly, can be set to operate the AH or WH system (for example, X10 address of House Code A) from 4 degree F with the command *SendAction(h, 'sendplc', 'a1 dim 0')* up to 127 degree F with the command *SendAction(h, 'sendplc', 'a4 dim 87')*.

4.3 Price_Naming_Assistant

The Price_Naming_Assistant menu can assist a consumer for his/her price naming decision makings for controlling of an appliance. It is helpful if a consumer can be informed of the trend of the approaching real-time prices. ERCOT, however, doesn't provide the Day-Ahead or forecasted real-time prices. Therefore, in our prototypes, the forecasted next 24-hour real-time prices are calculated using the previous 4 day data. If the current day is a weekday, the previous 4 weekdays of ERCOT Balancing Energy prices of North Zone are used. The previous 4 weekend prices are used instead if the current day is a weekend. These historical prices are averaged and smoothed by the method of Moving-Average before graphically plotted and compared with the current real-time price.

Please note that these calculated prices are only the estimated prices based on the historical data not from the congestion managements or real-time market settlements of utilities. Consequently, the predicted real-time prices may not be accurate. In addition, when the real-time pricing of electricity market is in effect, these prices should be provided by the utilities not by ERCOT, and different locations can have different real-time prices.

A consumer can set a deadline to operate an appliance. The minimum price between the current time and the deadline and the estimated time for that minimum price will be identified and illustrated onscreen. A consumer can have an option to set the named price as well. He/she still has to set the timer to update the real-time data before the load controller in this menu can start.

The load controller will compare the current real-time time price with the named price and the estimated minimum price. If the real-time price drops lower than either the named price or the estimated minimum price, the load controller will send the switch-on command to operate an appliance. The switch-on command is also sent when the estimated minimum time is reached.

With the guides from the Price_Naming_Assistant menu, the consumer can better manage his/her load consumptions for savings.

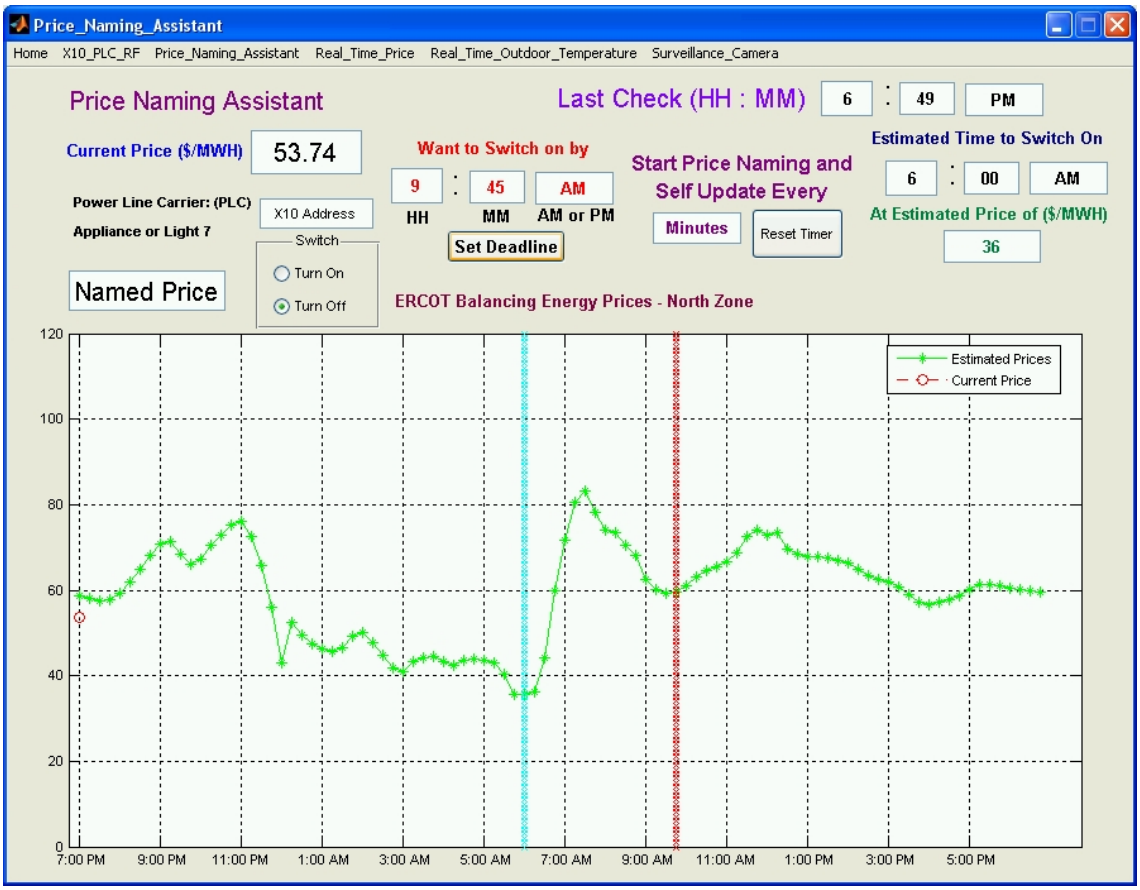


Figure 4.8 Print screen of the MATLAB customer's graphic user interface (GUI) in the Price_Naming_Assistant mode

4.4 Real_Time_Price Mode

It is anticipated that the real-time pricing data will be available in a public access website. A similar environment will be created in the experiment to simulate the real-time operating conditions. When a user selects the Real_Time_Price mode from the top menu, the MATLAB GUI will exhibit the real-time and the historical (from 12 AM of the current date) ERCOT wholesale balancing energy prices of the North Zone (Fig. 4.9). These data are assumed to be the retail prices that end-use customers in Arlington, TX perceive. These pseudo real-time pricing data are extracted from the HTLM source

codes of [53] using the MATLAB function *urlread*. When the real-time price based market is in operation in the near future, the price information should be provided by the local utilities, not from ERCOT, and different locations may have different rates due to possible congestion within the utility system.

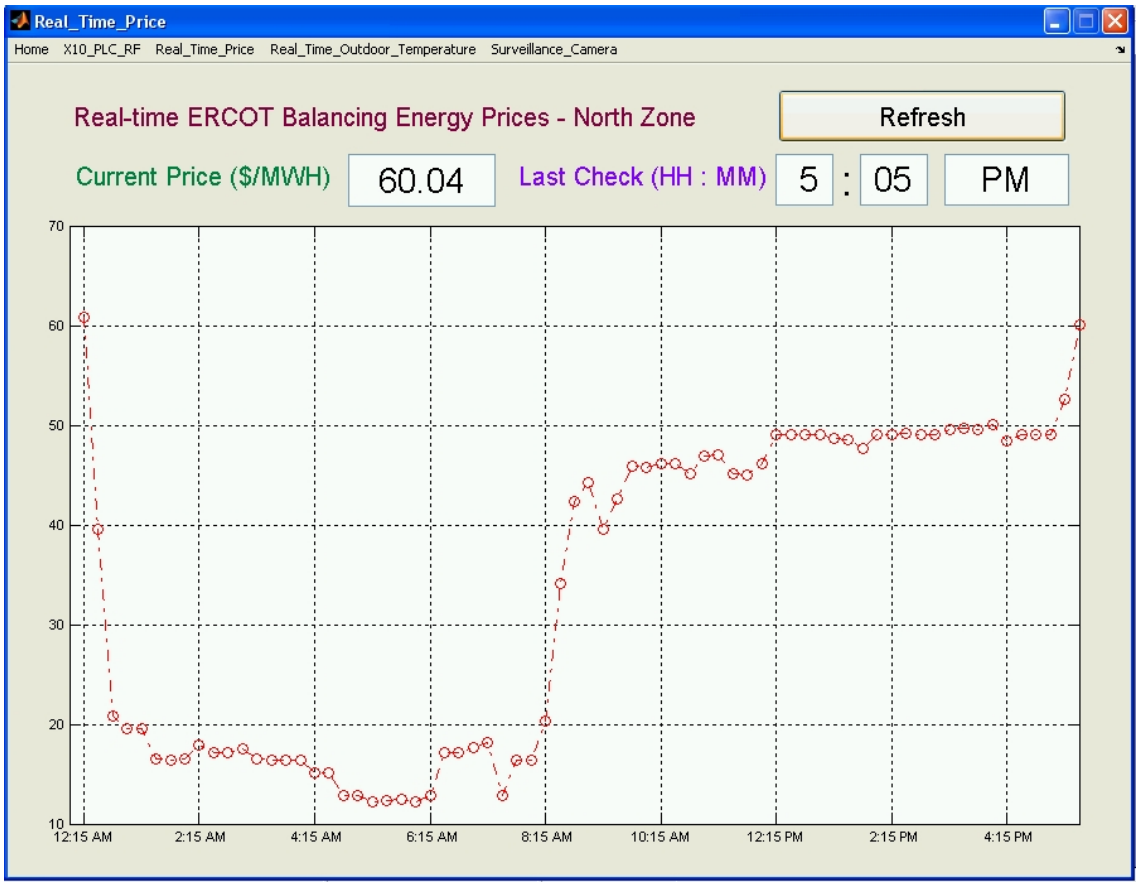


Figure 4.9 Print screen of the MATLAB customer's graphic user interface (GUI) in the Real_Time_Price mode

The GUI in Fig 4.9 shows the current price of $60.04/MWH at the current time at 5:05PM and the fluctuated prices since midnight of that date. This plot of time varying electricity prices illustrates the costs of load consumption if a customer chooses

to operate an appliance. An end-use customer can observe the trends of the real-time and historical prices to assist in his/her load management decision makings. A consumer can click “Refresh” button to update and retrieve the current updated data. The value of the current price in $/MWH is also the information presented onscreen of GUI in the X10_PLC_RF mode.

4.5 Real_Time_Outdoor_Temperature Mode

Electricity pricing is sensitive to temperatures. High consumption of air conditioner or heater can increase aggregate demand and drive up electricity price.

When a user selects the Real_Time_Outdoor_Temperature mode from the top menu, the MATLAB GUI will exhibit the real-time and the forecasted (next 48 hours) outdoor temperatures of Arlington, TX in degree F (Fig. 4.10). The data are extracted from the HTLM source codes of [54] using the MATLAB function *urlread*. The “Refresh” button is for a user to acquire the updated weather data.

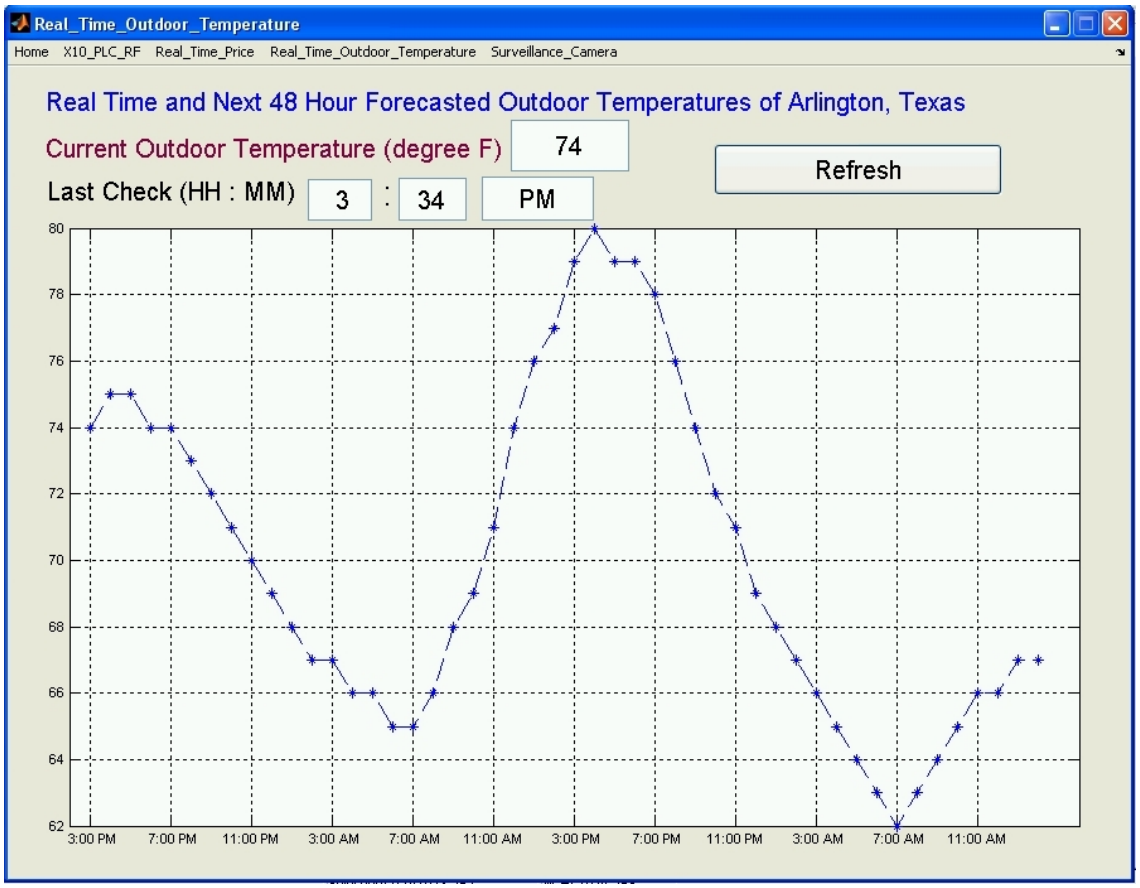


Figure 4.10 Print screen of the MATLAB customer's graphic user interface (GUI) in the Real_Time_Outdoor_Temperature mode

The GUI in Fig 4.10 shows the current outdoor temperature of 74 degree F of the current time at 3:34PM and the next 48 hour forecast data. A customer can graphically anticipate the coming weather condition to correctly set or adjust his/her operations of the thermostat set points of AH or WH, or other weather sensitive appliances.

Accompanying with the graphic real-time price, the graphic outdoor temperature information can assist a user for better load consumption managements.

The value of the current outdoor temperature is presented onscreen of GUI in the X10_PLC_RF mode as well.

4.6 Surveillance_Camera Mode

This menu provides a user a possibility to integrate other functions to the home automation system. In the prototype, surveillance camera for home security is included. In the Surveillance_Camera mode, the GUI initiates the MATLAB image processing toolbox [55] to acquire the image from connected surveillance camera and store in the video input object using the command *handles.vid = videoinput('winvideo')*. A user can pause, or resume the display preview by selecting the corresponding buttons (Fig. 4.11).

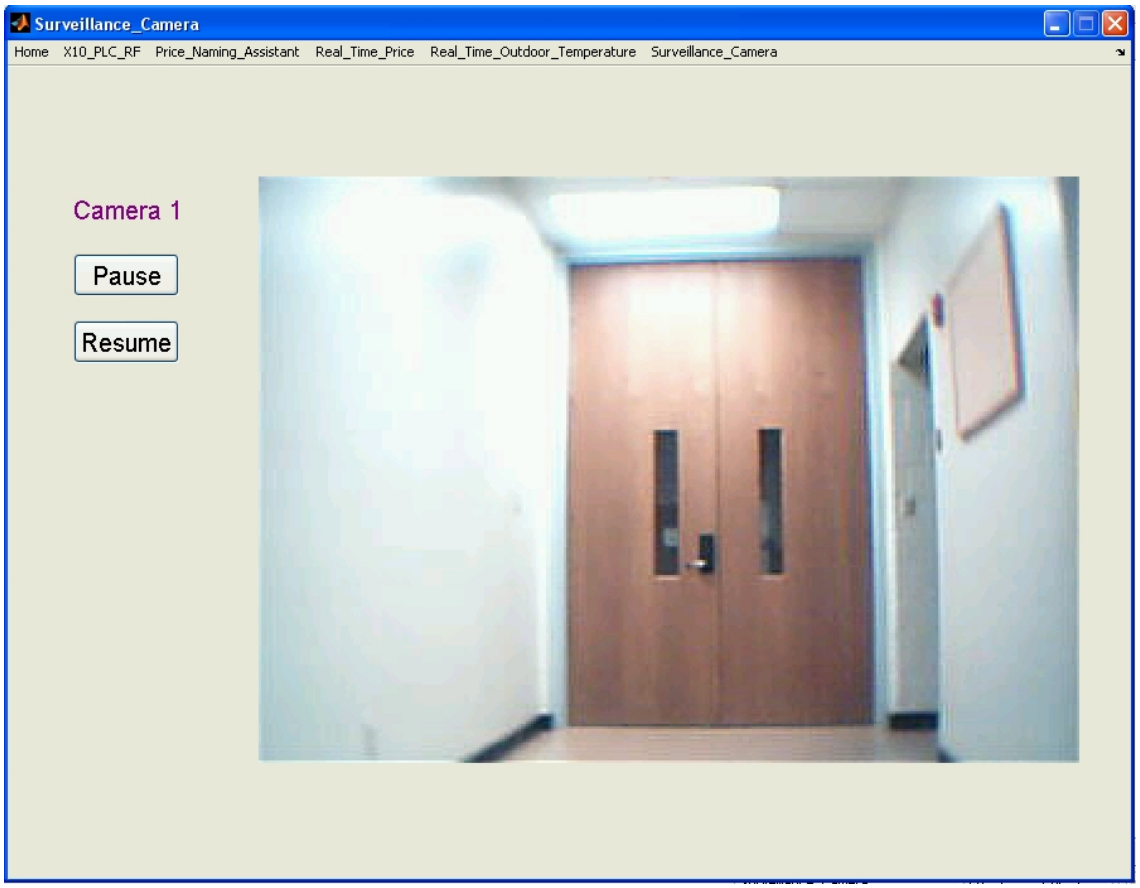


Figure 4.11 Print screen of the MATLAB customer's graphic user interface (GUI) in the Surveillance_Camera mode

CHAPTER 5

CONCLUSION

Electric deregulations can not be effectively and efficiently accomplished if the wholesale and retail electricity markets are disconnected. At current practice, the whole market supply side, on one hand, has time varying generating costs while the traditional retail market demand side, on the other hand however, receives flat rate monthly electrical bills.

Previous market failures and crises were so severe and were the evidences of the prerequisites to link both of these two markets. This dissertation, accordingly, realizes the vitality of demand response program to engage in this role.

A simple yet effective consumer-centered (contrasting to dated utility-centered) computational user interface for end-use residential consumers' load managements in real-time electricity pricing environment is introduced.

Supporting infrastructures to enhance demand response programs in various regional reliability councils are gradually increasing as ordered by the Energy Policy Act 2005. With these approaching developments, the dissertation's contributions can serve as tools for better electric deregulations.

5.1 Research Contributions

The major contributions of this research can be summarized as follows:

a) Possible solutions, ideas, and applications to link the wholesale utility and retail end-use residential customers for effective and efficient deregulated electricity market are provided.

b) An integrated user interface appliance control and home automation system to allow customer participation in controlling loads accordingly to the real time pricing, outdoor temperature info received is designed and implemented.

c) The expertise in both fields of demand side management and the home automation is integrated in this research's load control system. Other benefits of home automation, e.g., home security and surveillance system, are included as well.

d) The load control strategies (i.e., Steps of Temperatures load control strategy for controlling air conditioner/heater and water heater loads and Price Naming load control strategy for controlling washer/dryer and dish washer loads) are introduced.

e) The proposed system is residential customer centered, not utility centered. The end-use residential customers can use this GUI to control and manage their own appliance load consumptions according to their preferences (unlike the traditional direct load control demand response program, where utilities can cycle off their air conditioner/heater or shed loads when the generation or transmission capacities are stressed, for example).

f) MATLAB is experimented as a user interface platform for a residential customer's energy consumption decision making. Real-time pricing and outdoor temperature information are extracted and graphically exhibited to the customer using the MATLAB computations.

g) The "smart" end-use residential customers in the states of Illinois, for example, after the real-time pricing of electricity has been in effect since 2007, have to use their own judgments to shift their load consumptions to lower price periods for savings. This integrated system can be a tool to serve those consumers (4.5 million in Illinois alone) and to enhance the benefits of demand response program in real-time electricity pricing markets.

5.2 Possible Future Researches

This prototype has potentials to integrate other capabilities to enhance the capabilities of the system. Future researchers can use the prototype as a stepping stone and guideline for their further developments. Some possible add-ons are as follows:

a) Besides home automation protocols of X10 power line carrier and 300MHz radio frequency that implemented in the prototype, future researchers can employ other prospective protocols such as Z-wave [56] or Zigbee IEEE 802.15.4 standard [57]. These protocols are based on radio frequency meshed networking (in the US, Z-wave operates on 908.42MHz and Zigbee operates on 915 MHz). Mesh networking is reliable in that each equipment can serve as a repeater to carry messages from a source to a destination. Z-wave, however, is a proprietary protocol. Currently, to integrate Z-wave

to other software applications requires license from Zensys [58], and Zigbee protocol more concentrates on industrial rather than residential applications.

b) Include the electricity usage data from wattmeter for consumers and establish database management in storing historical and current interval data that the wattmeter logs.

c) Although the TXB16 X10 thermostat in the prototype can transmit indoor temperature data to display onscreen when queried, alternately other hardware can be integrated to send these data: USB thermometer [59] can be a possible candidate.

To transmit data from hardware to be executed in MATLAB computations and applications according to above sections a) to c) can be achieved by ActiveX COM interfaces. Presently, the Z-wave and Zigbee protocols, wattmeter, and USB thermostat in the markets do not support ActiveX COM interfaces. These barriers should be lessening in the near future for software developers and researchers.

d) A possible add-on feature to include in GUI function is web-service. This technology enables consumers to remotely (from anyplace in the globe) control their appliances, or access data (e.g., energy consumption, view the surveillance camera or security system) while away using internet.

e) Develop other load control strategies to implement in the GUI. A possibility is to apply project scheduling concepts from industrial engineering and specify each load as a task to satisfy resource constraints (e.g., energy, or time).

f) To extend the reach to the end-use consumers, future researchers may implement the integrated load control system to other platforms than MATLAB

environment, for example, National Instrument LabView [60], or, potentially, in handheld hardware devices such as Palm, or Apple's iPhone using its software developer kit (SDK) [61].

APPENDIX A

MATLAB USER INTERFACE SOUCE CODES – HOME

```
function varargout = Home(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Home_OpeningFcn, ...
                   'gui_OutputFcn',  @Home_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function Home_OpeningFcn(hObject, eventdata, handles, varargin)
    handles.LOGO = imread('logo1.jpg');
    image(handles.LOGO);

handles.output = hObject;
guidata(hObject, handles);

function varargout = Home_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function Home_Callback(hObject, eventdata, handles)
Home

function X10_PLC_RF_Callback(hObject, eventdata, handles)
X10_PLC_RF

function Price_Naming_Assistant_Callback...
    (hObject, eventdata, handles)
Price_Naming_Assistant

function Real_Time_Price_Callback(hObject, eventdata, handles)
Real_Time_Price

function Real_Time_Outdoor_Temperature_Callback...
    (hObject, eventdata, handles)

Real_Time_Outdoor_Temperature

function Surveillance_Camera_Callback(hObject, eventdata, handles)
Surveillance_Camera
```

APPENDIX B

MATLAB USER INTERFACE SOUCE CODES – X10_PLC_RF

```
function varargout = X10_PLC_RF(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @X10_PLC_RF_OpeningFcn, ...
                   'gui_OutputFcn',  @X10_PLC_RF_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function X10_PLC_RF_OpeningFcn(hObject, eventdata, handles, varargin)
    current_time_string=datestr(now,16);
    handles.current_time_string=current_time_string;
    set(handles.Current_Hour_Text,'String', ...
        handles.current_time_string(1:2));
    set(handles.Current_Minute_Text,'String', ...
        handles.current_time_string(4:5));
    set(handles.Current_AMPM_Text,'String', ...
        handles.current_time_string(7:8));
% Set real-time price edit text
        [north_zone_prices,num_location_time_frame,new_time_frame]=...
            ERCOT_prices;
        % show x tick only one every 2 hours
        length_north_zone_prices=length(north_zone_prices);
        current_north_zone_prices=north_zone_prices...
            (length_north_zone_prices);

    handles.north_zone_prices=north_zone_prices;
    handles.current_north_zone_prices=current_north_zone_prices;

    set(handles.Current_Price_Text,'String', ...
        num2str(handles.current_north_zone_prices));

    % plot weather forecast
        [Outdoor_Temperature, new_time_frame]=...
            outdoor_weather_from_findlocalweather;
    % pick the first element as the current temperature
        current_Outdoor_Temperature=Outdoor_Temperature(1);
    handles.Outdoor_Temperature=Outdoor_Temperature;
    handles.current_Outdoor_Temperature=current_Outdoor_Temperature;
    set(handles.current_Outdoor_Temperature_Text,'String',...
    num2str(handles.current_Outdoor_Temperature));

    handles.update_X10_PLC_RF=60*str2double(get...
        (handles.update_X10_PLC_RF_Text,'String'));

    handles.X10_address1=get(handles.X10_Address_Text1,'String');
    handles.X10_address2=get(handles.X10_Address_Text2,'String');
    handles.X10_address3=get(handles.X10_Address_Text3,'String');
    handles.X10_address4=get(handles.X10_Address_Text4,'String');
    handles.RF_address5=get(handles.RF_Address_Text5,'String');
```

```
handles.RF_address6=get(handles.RF_Address_Text6,'String');

handles.Named_Price1=str2double(get...
    (handles.Named_Price_Text1,'String'));
handles.Named_Price2=str2double(get...
    (handles.Named_Price_Text2,'String'));
handles.Named_Price3=str2double(get...
    (handles.Named_Price_Text3,'String'));
handles.Named_Price4=str2double(get...
    (handles.Named_Price_Text4,'String'));
handles.Named_Price5=str2double(get...
    (handles.Named_Price_Text5,'String'));
handles.Named_Price6=str2double(get...
    (handles.Named_Price_Text6,'String'));

handles.AH_Step_Temp1=get(handles.AH_Step_Temp_Text1,'String');
handles.AH_Step_Temp2=get(handles.AH_Step_Temp_Text2,'String');
handles.AH_Step_Temp3=get(handles.AH_Step_Temp_Text3,'String');
handles.AH_Step_Temp4=get(handles.AH_Step_Temp_Text4,'String');
handles.AH_Step_Temp5=get(handles.AH_Step_Temp_Text5,'String');

handles.AH_Step_Price1=get(handles.AH_Step_Price_Text1,'String');
handles.AH_Step_Price2=get(handles.AH_Step_Price_Text2,'String');
handles.AH_Step_Price3=get(handles.AH_Step_Price_Text3,'String');
handles.AH_Step_Price4=get(handles.AH_Step_Price_Text4,'String');
handles.AH_Step_Price5=get(handles.AH_Step_Price_Text5,'String');

handles.WH_Step_Temp1=get(handles.WH_Step_Temp_Text1,'String');
handles.WH_Step_Temp2=get(handles.WH_Step_Temp_Text2,'String');
handles.WH_Step_Temp3=get(handles.WH_Step_Temp_Text3,'String');
handles.WH_Step_Temp4=get(handles.WH_Step_Temp_Text4,'String');
handles.WH_Step_Temp5=get(handles.WH_Step_Temp_Text5,'String');

handles.WH_Step_Price1=get(handles.WH_Step_Price_Text1,'String');
handles.WH_Step_Price2=get(handles.WH_Step_Price_Text2,'String');
handles.WH_Step_Price3=get(handles.WH_Step_Price_Text3,'String');
handles.WH_Step_Price4=get(handles.WH_Step_Price_Text4,'String');
handles.WH_Step_Price5=get(handles.WH_Step_Price_Text5,'String');

if  handles.current_north_zone_prices <= handles.Named_Price1,
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address1 ' on']);
end

if  handles.current_north_zone_prices <= handles.Named_Price2,
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address2 ' on']);
end

if  handles.current_north_zone_prices <= handles.Named_Price3,
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address3 ' on']);
end

if  handles.current_north_zone_prices <= handles.Named_Price4,
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address4 ' on']);
end
```

```
if  handles.current_north_zone_prices <= handles.Named_Price5,
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendrf', [handles.RF_address5 ' on']);
end

if  handles.current_north_zone_prices <= handles.Named_Price6,
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendrf', [handles.RF_address6 ' on']);
end

h = actxserver('X10.ActiveHome.1');

handles.on_status1=SendAction(h, 'queryplc', ...
    [handles.X10_address1 ' on']);
if handles.on_status1==1,
    set(handles.Brightness_Percent_Slider1,'Value',100);
    set(handles.Brightness_Percent_Text1,'String', '100');
    set(handles.Turn_On_Button1,'Value',1);
    set(handles.Turn_Off_Button1,'Value',0);
else
    set(handles.Brightness_Percent_Slider1,'Value',0);
    set(handles.Brightness_Percent_Text1,'String', '0');
    set(handles.Turn_On_Button1,'Value',0);
    set(handles.Turn_Off_Button1,'Value',1);
end

handles.on_status2=SendAction(h, 'queryplc', ...
    [handles.X10_address2 ' on']);
if handles.on_status2==1,
    set(handles.Brightness_Percent_Slider2,'Value',100);
    set(handles.Brightness_Percent_Text2,'String', '100');
    set(handles.Turn_On_Button2,'Value',1);
    set(handles.Turn_Off_Button2,'Value',0);
else
    set(handles.Brightness_Percent_Slider2,'Value',0);
    set(handles.Brightness_Percent_Text2,'String', '0');
    set(handles.Turn_On_Button2,'Value',0);
    set(handles.Turn_Off_Button2,'Value',1);
end

handles.on_status3=SendAction(h, 'queryplc', ...
    [handles.X10_address3 ' on']);
if handles.on_status3==1,
    set(handles.Brightness_Percent_Slider3,'Value',100);
    set(handles.Brightness_Percent_Text3,'String', '100');
    set(handles.Turn_On_Button3,'Value',1);
    set(handles.Turn_Off_Button3,'Value',0);
else
    set(handles.Brightness_Percent_Slider3,'Value',0);
    set(handles.Brightness_Percent_Text3,'String', '0');
    set(handles.Turn_On_Button3,'Value',0);
    set(handles.Turn_Off_Button3,'Value',1);
end

handles.on_status4=SendAction(h, 'queryplc', ...
    [handles.X10_address4 ' on']);
if handles.on_status4==1,
    set(handles.Brightness_Percent_Slider4,'Value',100);
    set(handles.Brightness_Percent_Text4,'String', '100');
```

```
        set(handles.Turn_On_Button4,'Value',1);
        set(handles.Turn_Off_Button4,'Value',0);
    else
        set(handles.Brightness_Percent_Slider4,'Value',0);
        set(handles.Brightness_Percent_Text4,'String', '0');
        set(handles.Turn_On_Button4,'Value',0);
        set(handles.Turn_Off_Button4,'Value',1);
    end

% Operate AH at Temperature 1
if (handles.AH_Step_Price2>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.AH_Step_Price1),
    AH_set_temperature=handles.AH_Step_Temp1;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(AH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.AH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate AH at Temperature 2
if (handles.AH_Step_Price3>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.AH_Step_Price2),
    AH_set_temperature=handles.AH_Step_Temp2;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(AH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.AH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate AH at Temperature 3
if (handles.AH_Step_Price4>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.AH_Step_Price3),
    AH_set_temperature=handles.AH_Step_Temp3;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(AH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.AH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate AH at Temperature 4
if (handles.AH_Step_Price5>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.AH_Step_Price4),
    AH_set_temperature=handles.AH_Step_Temp4;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(AH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.AH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate AH at Temperature 5
if (handles.current_north_zone_prices>handles.AH_Step_Price5),
    AH_set_temperature=handles.AH_Step_Temp5;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(AH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
```

```
    SendAction(h, 'sendplc', [handles.AH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate WH at Temperature 5
if (handles.WH_Step_Price2>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.WH_Step_Price1),
    WH_set_temperature=handles.WH_Step_Temp5;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(WH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.WH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate WH at Temperature 4
if (handles.WH_Step_Price3>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.WH_Step_Price2),
    WH_set_temperature=handles.WH_Step_Temp4;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(WH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.WH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate WH at Temperature 3
if (handles.WH_Step_Price4>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.WH_Step_Price3),
    WH_set_temperature=handles.WH_Step_Temp3;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(WH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.WH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate WH at Temperature 2
if (handles.WH_Step_Price5>=handles.current_north_zone_prices) & ...
    (handles.current_north_zone_prices>handles.WH_Step_Price4),
    WH_set_temperature=handles.WH_Step_Temp2;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(WH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.WH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Operate WH at Temperature 1
if (handles.current_north_zone_prices>handles.WH_Step_Price5),
    WH_set_temperature=handles.WH_Step_Temp1;
    [step_dim_level,unit_number,preset_level]=...
        thermostat_dim_level_decode(WH_set_temperature);
    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.WH_Address1, ...
        num2str(unit_number), ' dim ', num2str(step_dim_level)]);
end,

% Choose default command line output for X10_PLC_RF
```

```
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

function varargout = X10_PLC_RF_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;

function update_X10_PLC_RF_Text_Callback(hObject, eventdata, handles)

% get current data
% Change from seconds to minutes

    handles.update_X10_PLC_RF=60*str2double(get...
        (handles.update_X10_PLC_RF_Text,'String'));

t = timer('Period', handles.update_X10_PLC_RF, ...
        'ExecutionMode','fixedSpacing', ...
        'StartFcn', {'X10_PLC_RF'}, ...
        'TimerFcn', {'X10_PLC_RF'})

start(t)

    guidata(hObject, handles);

function update_X10_PLC_RF_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function update_X10_PLC_RF_Text_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.update_X10_PLC_RF_Text,'String','');

function update_X10_PLC_RF_Text_KeyPressFcn...
        (hObject, eventdata, handles)

    set(handles.update_X10_PLC_RF_Text,'String','');

function Brightness_Percent_Slider1_Callback...
        (hObject, eventdata, handles)

    handles.slider_value1=round(get...
        (handles.Brightness_Percent_Slider1,'Value'));
    set(handles.Brightness_Percent_Text1,'String', ...
        num2str(handles.slider_value1));

    if handles.slider_value1==0,
        set(handles.Turn_On_Button1,'Value',0);
        set(handles.Turn_Off_Button1,'Value',1);
    else
        set(handles.Turn_On_Button1,'Value',1);
        set(handles.Turn_Off_Button1,'Value',0);
    end
```

```
    h = actxserver('X10.ActiveHome.1');
    handles.dim_status1=SendAction(h, 'queryplc', ...
        [handles.X10_address1 ' dim']);
    dim_different1=handles.slider_value1-handles.dim_status1;
    if handles.dim_status1 > handles.slider_value1,
        SendAction(h, 'sendplc', [handles.X10_address1, ...
            ' dim ', num2str(abs(dim_different1))]);
    else
        SendAction(h, 'sendplc', [handles.X10_address1, ...
            ' bright ', num2str(abs(dim_different1))]);
    end
    guidata(hObject, handles);

function Brightness_Percent_Slider1_CreateFcn...
        (hObject, eventdata, handles)

if isequal(get(hObject,'BackgroundColor'), get(0,...
        'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function X10_Address_Text1_Callback(hObject, eventdata, handles)

    handles.X10_address1=get(handles.X10_Address_Text1,'String');
    guidata(hObject, handles);

function X10_Address_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

    function X10_Address_Text1_ButtonDownFcn...
            (hObject, eventdata, handles)

    set(handles.X10_Address_Text1,'String','');

function X10_Address_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text1,'String','');

function Brightness_Percent_Text1_Callback...
        (hObject, eventdata, handles)

    value_Brightness_Percent_Text1 = str2double(get...
        (handles.Brightness_Percent_Text1,'String'));
    % Determine whether val_Brightness_Percent_Text1
    % is a number between 0 and 100
    if isnumeric(value_Brightness_Percent_Text1) & ...
        length(value_Brightness_Percent_Text1)==1 & ...
        value_Brightness_Percent_Text1 >= get...
        (handles.Brightness_Percent_Slider1,'Min') & ...
        value_Brightness_Percent_Text1 <= get...
        (handles.Brightness_Percent_Slider1,'Max'),
        set(handles.Brightness_Percent_Slider1,'Value',...
            value_Brightness_Percent_Text1);
    else
    % Increment the error count, and display it
```

```
        set(handles.Brightness_Percent_Text1,'String', ...
            'Enter 0-100 only');
    end

    if value_Brightness_Percent_Text1 == 0,
        set(handles.Turn_On_Button1,'Value',0);
        set(handles.Turn_Off_Button1,'Value',1);
    end
    if value_Brightness_Percent_Text1>0 & 100>=...
            value_Brightness_Percent_Text1,
        set(handles.Turn_On_Button1,'Value',1);
        set(handles.Turn_Off_Button1,'Value',0);
    end
    h = actxserver('X10.ActiveHome.1');
    handles.dim_status1=SendAction(h, 'queryplc', ...
        [handles.X10_address1 ' dim']);
    dim_different1=value_Brightness_Percent_Text1-handles.dim_status1;
    if handles.dim_status1 > value_Brightness_Percent_Text1,

        SendAction(h, 'sendplc', [handles.X10_address1, ...
            ' dim ', num2str(abs(dim_different1))]);
    else
        SendAction(h, 'sendplc', [handles.X10_address1, ...
            ' bright ', num2str(abs(dim_different1))]);
    end
    guidata(hObject, handles);

function Brightness_Percent_Text1_CreateFcn(hObject, ...
        eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Turn_On_Button1_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');

    % to make sure the initial state is not already
    % turn on and in dim mode to prevent conflict
    % with slider dim command.

    SendAction(h, 'sendplc', [handles.X10_address1 ' on']);
    set(handles.Brightness_Percent_Slider1,'Value',100);
    set(handles.Brightness_Percent_Text1,'String', '100');

function Turn_Off_Button1_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address1 ' off']);
    set(handles.Brightness_Percent_Slider1,'Value',0);
    set(handles.Brightness_Percent_Text1,'String', '0');

function Brightness_Percent_Text1_ButtonDownFcn(...
        hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text1,'String','');
```

```
function Brightness_Percent_Text1_KeyPressFcn(...
        hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text1,'String','');

function Named_Price_Text1_Callback(hObject, eventdata, handles)

    handles.Named_Price1=str2double(get...
        (handles.Named_Price_Text1,'String'));
    guidata(hObject, handles);

function Named_Price_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Home_Callback(hObject, eventdata, handles)
Home

function X10_PLC_RF_Callback(hObject, eventdata, handles)
X10_PLC_RF

function Price_Naming_Assistant_Callback...
       (hObject, eventdata, handles)
Price_Naming_Assistant

function Real_Time_Price_Callback(hObject, eventdata, handles)
Real_Time_Price

function Real_Time_Outdoor_Temperature_Callback(...
        hObject, eventdata, handles)
Real_Time_Outdoor_Temperature

function Surveillance_Camera_Callback(hObject, eventdata, handles)
Surveillance_Camera

function Current_Price_Text_Callback(hObject, eventdata, handles)

function Current_Price_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Hour_Text_Callback(hObject, eventdata, handles)

function Current_Hour_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Minute_Text_Callback(hObject, eventdata, handles)
```

```
function Current_Minute_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Current_AMPM_Text_Callback(hObject, eventdata, handles)

function Current_AMPM_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function current_Outdoor_Temperature_Text_Callback(...
        hObject, eventdata, handles)

function current_Outdoor_Temperature_Text_CreateFcn(...
        hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Brightness_Percent_Slider2_Callback...
        (hObject, eventdata, handles)

    handles.slider_value2=round(get...
        (handles.Brightness_Percent_Slider2,'Value'));
    set(handles.Brightness_Percent_Text2,'String', ...
        num2str(handles.slider_value2));

    if handles.slider_value2==0,
        set(handles.Turn_On_Button2,'Value',0);
        set(handles.Turn_Off_Button2,'Value',1);
    else
        set(handles.Turn_On_Button2,'Value',1);
        set(handles.Turn_Off_Button2,'Value',0);
    end

    h = actxserver('X10.ActiveHome.1');
    handles.dim_status2=SendAction(h, 'queryplc', ...
        [handles.X10_address2 ' dim']);
    dim_different2=handles.slider_value2-handles.dim_status2;
    if handles.dim_status2 > handles.slider_value2,

        SendAction(h, 'sendplc', [handles.X10_address2, ...
            ' dim ', num2str(abs(dim_different2))]);

    else
        SendAction(h, 'sendplc', [handles.X10_address2, ...
            ' bright ', num2str(abs(dim_different2))]);
    end
    guidata(hObject, handles);
```

```
function Brightness_Percent_Slider2_CreateFcn...
        (hObject, eventdata, handles)

if isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function X10_Address_Text2_Callback(hObject, eventdata, handles)

    handles.X10_address2=get(handles.X10_Address_Text2,'String');
    guidata(hObject, handles);

function X10_Address_Text2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function X10_Address_Text2_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text2,'String','');

function X10_Address_Text2_KeyPressFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text2,'String','');

function Brightness_Percent_Text2_Callback...
        (hObject, eventdata, handles)

    value_Brightness_Percent_Text2 = str2double...
        (get(handles.Brightness_Percent_Text2,'String'));
    % Determine whether val_Brightness_Percent_Text2
    % is a number between 0 and 100
    if isnumeric(value_Brightness_Percent_Text2) & ...
            length(value_Brightness_Percent_Text2)==1 & ...
        value_Brightness_Percent_Text2 >= get...
        (handles.Brightness_Percent_Slider2,'Min') & ...
        value_Brightness_Percent_Text2 <= get...
        (handles.Brightness_Percent_Slider2,'Max'),
        set(handles.Brightness_Percent_Slider2,...
            'Value',value_Brightness_Percent_Text2);
    else
    % Increment the error count, and display it
        set(handles.Brightness_Percent_Text2,'String',...
            'Enter 0-100 only');
    end

    if value_Brightness_Percent_Text2 == 0,
        set(handles.Turn_On_Button2,'Value',0);
        set(handles.Turn_Off_Button2,'Value',1);
    end
    if value_Brightness_Percent_Text2>0 & 100>=...
            value_Brightness_Percent_Text2,
        set(handles.Turn_On_Button2,'Value',1);
        set(handles.Turn_Off_Button2,'Value',0);
    end
```

```
    h = actxserver('X10.ActiveHome.1');
    handles.dim_status2=SendAction(h, 'queryplc', ...
        [handles.X10_address2 ' dim']);
    dim_different2=value_Brightness_Percent_Text2-...
        handles.dim_status2;
    if handles.dim_status2 > value_Brightness_Percent_Text2,

        SendAction(h, 'sendplc', [handles.X10_address2, ...
            ' dim ', num2str(abs(dim_different2))]);
    else
        SendAction(h, 'sendplc', [handles.X10_address2, ...
            ' bright ', num2str(abs(dim_different2))]);
    end
    guidata(hObject, handles);

function Brightness_Percent_Text2_CreateFcn...
        (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), get...
        (0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Turn_On_Button2_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');

    % to make sure the initial state is not already
    % turn on and in dim mode
    % prevent conflict with slider dim command.

    SendAction(h, 'sendplc', [handles.X10_address2 ' on']);
    set(handles.Brightness_Percent_Slider2,'Value',100);
    set(handles.Brightness_Percent_Text2,'String', '100');

function Turn_Off_Button2_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');

    SendAction(h, 'sendplc', [handles.X10_address2 ' off']);
    set(handles.Brightness_Percent_Slider2,'Value',0);
    set(handles.Brightness_Percent_Text2,'String', '0');

function Brightness_Percent_Text2_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text2,'String','');

function Brightness_Percent_Text2_KeyPressFcn...
        (hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text2,'String','');

function Named_Price_Text2_Callback(hObject, eventdata, handles)

    handles.Named_Price2=str2double(get...
        (handles.Named_Price_Text2,'String'));
    guidata(hObject, handles);
```

```
function Named_Price_Text2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Brightness_Percent_Slider3_Callback...
        (hObject, eventdata, handles)

    handles.slider_value3=round(get...
        (handles.Brightness_Percent_Slider3,'Value'));
    set(handles.Brightness_Percent_Text3,'String', ...
        num2str(handles.slider_value3));

    if handles.slider_value3==0,
        set(handles.Turn_On_Button3,'Value',0);
        set(handles.Turn_Off_Button3,'Value',1);
    else
        set(handles.Turn_On_Button3,'Value',1);
        set(handles.Turn_Off_Button3,'Value',0);
    end

    h = actxserver('X10.ActiveHome.1');
    handles.dim_status3=SendAction(h, 'queryplc', ...
        [handles.X10_address3 ' dim']);
    dim_different3=handles.slider_value3-handles.dim_status3;
    if handles.dim_status3 > handles.slider_value3,

        SendAction(h, 'sendplc', [handles.X10_address3, ...
            ' dim ', num2str(abs(dim_different3))]);

    else
        SendAction(h, 'sendplc', [handles.X10_address3, ...
            ' bright ', num2str(abs(dim_different3))]);
    end
    guidata(hObject, handles);


function Brightness_Percent_Slider3_CreateFcn...
        (hObject, eventdata, handles)

if isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function X10_Address_Text3_Callback(hObject, eventdata, handles)

    handles.X10_address3=get(handles.X10_Address_Text3,'String');
    guidata(hObject, handles);


function X10_Address_Text3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function X10_Address_Text3_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text3,'String','');


function X10_Address_Text3_KeyPressFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text3,'String','');

function Brightness_Percent_Text3_Callback...
        (hObject, eventdata, handles)

    value_Brightness_Percent_Text3 = str2double(get...
        (handles.Brightness_Percent_Text3,'String'));

    if isnumeric(value_Brightness_Percent_Text3) & ...
            length(value_Brightness_Percent_Text3)==1 & ...
        value_Brightness_Percent_Text3 >= get...
        (handles.Brightness_Percent_Slider3,'Min') & ...
        value_Brightness_Percent_Text3 <= get...
        (handles.Brightness_Percent_Slider3,'Max'),
        set(handles.Brightness_Percent_Slider3,...
            'Value',value_Brightness_Percent_Text3);
    else
    % Increment the error count, and display it
        set(handles.Brightness_Percent_Text3,'String', 'Enter 0-100 only');
    end

    if value_Brightness_Percent_Text3 == 0,
        set(handles.Turn_On_Button3,'Value',0);
        set(handles.Turn_Off_Button3,'Value',1);
    end
    if value_Brightness_Percent_Text3>0 & 100>=...
            value_Brightness_Percent_Text3,
        set(handles.Turn_On_Button3,'Value',1);
        set(handles.Turn_Off_Button3,'Value',0);
    end
    h = actxserver('X10.ActiveHome.1');
    handles.dim_status3=SendAction(h, 'queryplc', ...
        [handles.X10_address3 ' dim']);
    dim_different3=value_Brightness_Percent_Text3-handles.dim_status3;
    if handles.dim_status3 > value_Brightness_Percent_Text3,

        SendAction(h, 'sendplc', [handles.X10_address3, ...
            ' dim ', num2str(abs(dim_different3))]);
    else
        SendAction(h, 'sendplc', [handles.X10_address3, ...
            ' bright ', num2str(abs(dim_different3))]);
    end
    guidata(hObject, handles);

function Brightness_Percent_Text3_CreateFcn...
        (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Turn_On_Button3_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');

    % to make sure the initial state is not already
    % turn on and in dim mode
    % prevent conflict with slider dim command.

    SendAction(h, 'sendplc', [handles.X10_address3 ' on']);
    set(handles.Brightness_Percent_Slider3,'Value',100);
    set(handles.Brightness_Percent_Text3,'String', '100');
    %     end

function Turn_Off_Button3_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address3 ' off']);
    set(handles.Brightness_Percent_Slider3,'Value',0);
    set(handles.Brightness_Percent_Text3,'String', '0');

function Brightness_Percent_Text3_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text3,'String','');

function Brightness_Percent_Text3_KeyPressFcn...
        (hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text3,'String','');

function Named_Price_Text3_Callback(hObject, eventdata, handles)

    handles.Named_Price3=str2double(get...
        (handles.Named_Price_Text3,'String'));
    guidata(hObject, handles);

function Named_Price_Text3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Brightness_Percent_Slider4_Callback...
        (hObject, eventdata, handles)

    handles.slider_value4=round(get...
        (handles.Brightness_Percent_Slider4,'Value'));
    set(handles.Brightness_Percent_Text4,'String', ...
        num2str(handles.slider_value4));

    if handles.slider_value4==0,
        set(handles.Turn_On_Button4,'Value',0);
        set(handles.Turn_Off_Button4,'Value',1);
    else
        set(handles.Turn_On_Button4,'Value',1);
        set(handles.Turn_Off_Button4,'Value',0);
```

```
    end

    h = actxserver('X10.ActiveHome.1');
    handles.dim_status4=SendAction(h, 'queryplc', ...
        [handles.X10_address4 ' dim']);
    dim_different4=handles.slider_value4-handles.dim_status4;
    if handles.dim_status4 > handles.slider_value4,

        SendAction(h, 'sendplc', [handles.X10_address4, ...
            ' dim ', num2str(abs(dim_different4))]);

    else
        SendAction(h, 'sendplc', [handles.X10_address4, ...
            ' bright ', num2str(abs(dim_different4))]);
    end
    guidata(hObject, handles);


function Brightness_Percent_Slider4_CreateFcn...
        (hObject, eventdata, handles)

if isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

function X10_Address_Text4_Callback(hObject, eventdata, handles)

    handles.X10_address4=get(handles.X10_Address_Text4,'String');
    guidata(hObject, handles);

function X10_Address_Text4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function X10_Address_Text4_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text4,'String','');

function X10_Address_Text4_KeyPressFcn(hObject, eventdata, handles)

    set(handles.X10_Address_Text4,'String','');

function Brightness_Percent_Text4_Callback...
        (hObject, eventdata, handles)

    value_Brightness_Percent_Text4 = str2double(get...
        (handles.Brightness_Percent_Text4,'String'));
    % Determine whether val_Brightness_Percent_Text4
    % is a number between 0 and 100
    if isnumeric(value_Brightness_Percent_Text4) & ...
            length(value_Brightness_Percent_Text4)==1 & ...
        value_Brightness_Percent_Text4 >= get...
        (handles.Brightness_Percent_Slider4,'Min') & ...
        value_Brightness_Percent_Text4 <= get...
        (handles.Brightness_Percent_Slider4,'Max'),
```

```
        set(handles.Brightness_Percent_Slider4,'Value',...
            value_Brightness_Percent_Text4);
    else
    % Increment the error count, and display it
        set(handles.Brightness_Percent_Text4,...
            'String', 'Enter 0-100 only');
    end

    if value_Brightness_Percent_Text4 == 0,
        set(handles.Turn_On_Button4,'Value',0);
        set(handles.Turn_Off_Button4,'Value',1);
    end
    if value_Brightness_Percent_Text4>0 & 100>=...
            value_Brightness_Percent_Text4,
        set(handles.Turn_On_Button4,'Value',1);
        set(handles.Turn_Off_Button4,'Value',0);
    end
    h = actxserver('X10.ActiveHome.1');
    handles.dim_status4=SendAction(h, 'queryplc', ...
        [handles.X10_address4 ' dim']);
    dim_different4=value_Brightness_Percent_Text4-...
        handles.dim_status4;
    if handles.dim_status4 > value_Brightness_Percent_Text4,

        SendAction(h, 'sendplc', [handles.X10_address4, ...
            ' dim ', num2str(abs(dim_different4))]);
    else
        SendAction(h, 'sendplc', [handles.X10_address4, ...
            ' bright ', num2str(abs(dim_different4))]);
    end
    guidata(hObject, handles);

function Brightness_Percent_Text4_CreateFcn...
        (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Turn_On_Button4_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    % to make sure the initial state is not already
    % turn on and in dim mode
    % prevent conflict with slider dim command.
    SendAction(h, 'sendplc', [handles.X10_address4 ' on']);
    set(handles.Brightness_Percent_Slider4,'Value',100);
    set(handles.Brightness_Percent_Text4,'String', '100');

function Turn_Off_Button4_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendplc', [handles.X10_address4 ' off']);
    set(handles.Brightness_Percent_Slider4,'Value',0);
    set(handles.Brightness_Percent_Text4,'String', '0');

function Brightness_Percent_Text4_ButtonDownFcn...
        (hObject, eventdata, handles)
```

```
    set(handles.Brightness_Percent_Text4,'String','');

function Brightness_Percent_Text4_KeyPressFcn...
        (hObject, eventdata, handles)

    set(handles.Brightness_Percent_Text4,'String','');

function Named_Price_Text4_Callback(hObject, eventdata, handles)

    handles.Named_Price4=str2double(get...
        (handles.Named_Price_Text4,'String'));
    guidata(hObject, handles);

function Named_Price_Text4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function RF_Address_Text5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function RF_Address_Text5_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.RF_Address_Text5,'String','');

function RF_Address_Text5_KeyPressFcn(hObject, eventdata, handles)

    set(handles.RF_Address_Text5,'String','');

function RF_Address_Text5_Callback(hObject, eventdata, handles)

    handles.RF_Address5=get(handles.RF_Address_Text5,'String');
    guidata(hObject, handles);

function Turn_On_Button5_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendrf', [handles.RF_Address5 ' on']);

function Turn_Off_Button5_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendrf', [handles.RF_Address5 ' off']);

function Named_Price_Text5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Named_Price_Text5_Callback(hObject, eventdata, handles)
```

```
function RF_Address_Text6_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function RF_Address_Text6_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.RF_Address_Text6,'String','');

function RF_Address_Text6_KeyPressFcn(hObject, eventdata, handles)

    set(handles.RF_Address_Text6,'String','');

function RF_Address_Text6_Callback(hObject, eventdata, handles)

    handles.RF_Address6=get(handles.RF_Address_Text6,'String');
    guidata(hObject, handles);

function Turn_On_Button6_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendrf', [handles.RF_Address6 ' on']);

function Turn_Off_Button6_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    SendAction(h, 'sendrf', [handles.RF_Address6 ' off']);

function Named_Price_Text6_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Named_Price_Text6_Callback(hObject, eventdata, handles)

function AH_Address_Text1_Callback(hObject, eventdata, handles)

    handles.AH_Address1=get(handles.AH_Address_Text1,'String');
    guidata(hObject, handles);

function AH_Address_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Address_Text1_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.AH_Address_Text1,'String','');

function AH_Address_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Address_Text1,'String','');
```

```
function AH_Step_Temp_Text1_Callback(hObject, eventdata, handles)

    handles.AH_Step_Temp1=get(handles.AH_Step_Temp_Text1,'String');
    guidata(hObject, handles);

function AH_Step_Temp_Text1_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text1,'String','');

function AH_Step_Temp_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text1,'String','');

function AH_Step_Temp_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function AH_Step_Temp_Text2_Callback(hObject, eventdata, handles)

    handles.AH_Step_Temp2=get(handles.AH_Step_Temp_Text2,'String');
    guidata(hObject, handles);

function AH_Step_Temp_Text2_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text2,'String','');

function AH_Step_Temp_Text2_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text2,'String','');

function AH_Step_Temp_Text2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Temp_Text3_Callback(hObject, eventdata, handles)

    handles.AH_Step_Temp3=get(handles.AH_Step_Temp_Text3,'String');
    guidata(hObject, handles);

function AH_Step_Temp_Text3_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text3,'String','');

function AH_Step_Temp_Text3_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text3,'String','');

function AH_Step_Temp_Text3_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Temp_Text4_Callback(hObject, eventdata, handles)

    handles.AH_Step_Temp4=get(handles.AH_Step_Temp_Text4,'String');
    guidata(hObject, handles);

function AH_Step_Temp_Text4_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text4,'String','');

function AH_Step_Temp_Text4_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text4,'String','');

function AH_Step_Temp_Text4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Temp_Text5_Callback(hObject, eventdata, handles)

    handles.AH_Step_Temp5=get(handles.AH_Step_Temp_Text5,'String');
    guidata(hObject, handles);

function AH_Step_Temp_Text5_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text5,'String','');

function AH_Step_Temp_Text5_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Temp_Text5,'String','');

function AH_Step_Temp_Text5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Price_Text1_Callback(hObject, eventdata, handles)

    handles.AH_Step_Price1=get(handles.AH_Step_Price_Text1,'String');
    guidata(hObject, handles);

function AH_Step_Price_Text1_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text1,'String','');

function AH_Step_Price_Text1_KeyPressFcn(hObject, eventdata, handles)
```

```
    set(handles.AH_Step_Price_Text1,'String','');

function AH_Step_Price_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Price_Text2_Callback(hObject, eventdata, handles)

    handles.AH_Step_Price2=get(handles.AH_Step_Price_Text2,'String');
    guidata(hObject, handles);

function AH_Step_Price_Text2_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text2,'String','');

function AH_Step_Price_Text2_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text2,'String','');

function AH_Step_Price_Text2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Price_Text3_Callback(hObject, eventdata, handles)

    handles.AH_Step_Price3=get(handles.AH_Step_Price_Text3,'String');
    guidata(hObject, handles);

function AH_Step_Price_Text3_ButtonDownFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text3,'String','');

function AH_Step_Price_Text3_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text3,'String','');

function AH_Step_Price_Text3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Price_Text4_Callback(hObject, eventdata, handles)

    handles.AH_Step_Price4=get(handles.AH_Step_Price_Text4,'String');
    guidata(hObject, handles);

function AH_Step_Price_Text4_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text4,'String','');
```

```
function AH_Step_Price_Text4_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text4,'String','');

function AH_Step_Price_Text4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function AH_Step_Price_Text5_Callback(hObject, eventdata, handles)

    handles.AH_Step_Price5=get(handles.AH_Step_Price_Text5,'String');
    guidata(hObject, handles);

function AH_Step_Price_Text5_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text5,'String','');

function AH_Step_Price_Text5_KeyPressFcn(hObject, eventdata, handles)

    set(handles.AH_Step_Price_Text5,'String','');

function AH_Step_Price_Text5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Address_Text1_Callback(hObject, eventdata, handles)

    handles.WH_Address1=get(handles.WH_Address_Text1,'String');
    guidata(hObject, handles);

function WH_Address_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Address_Text1_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Address_Text1,'String','');

function WH_Address_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Address_Text1,'String','');

function WH_Step_Temp_Text1_Callback(hObject, eventdata, handles)

    handles.WH_Step_Temp1=get(handles.WH_Step_Temp_Text1,'String');
    guidata(hObject, handles);

function WH_Step_Temp_Text1_ButtonDownFcn...
```

```
        (hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text1,'String','');

function WH_Step_Temp_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text1,'String','');

function WH_Step_Temp_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Temp_Text2_Callback(hObject, eventdata, handles)

    handles.WH_Step_Temp2=get(handles.WH_Step_Temp_Text2,'String');
    guidata(hObject, handles);

function WH_Step_Temp_Text2_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text2,'String','');

function WH_Step_Temp_Text2_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text2,'String','');

function WH_Step_Temp_Text2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Temp_Text3_Callback(hObject, eventdata, handles)

    handles.WH_Step_Temp3=get(handles.WH_Step_Temp_Text3,'String');
    guidata(hObject, handles);


function WH_Step_Temp_Text3_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text3,'String','');

function WH_Step_Temp_Text3_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text3,'String','');

function WH_Step_Temp_Text3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Temp_Text4_Callback(hObject, eventdata, handles)
```

```
    handles.WH_Step_Temp4=get(handles.WH_Step_Temp_Text4,'String');
    guidata(hObject, handles);

function WH_Step_Temp_Text4_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text4,'String','');

function WH_Step_Temp_Text4_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text4,'String','');

function WH_Step_Temp_Text4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Temp_Text5_Callback(hObject, eventdata, handles)

    handles.WH_Step_Temp5=get(handles.WH_Step_Temp_Text5,'String');
    guidata(hObject, handles);

function WH_Step_Temp_Text5_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text5,'String','');

function WH_Step_Temp_Text5_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Temp_Text5,'String','');

function WH_Step_Temp_Text5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Price_Text1_Callback(hObject, eventdata, handles)

    handles.WH_Step_Price1=get(handles.WH_Step_Price_Text1,'String');
    guidata(hObject, handles);

function WH_Step_Price_Text1_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text1,'String','');

function WH_Step_Price_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text1,'String','');

function WH_Step_Price_Text1_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end


function WH_Step_Price_Text2_Callback(hObject, eventdata, handles)

    handles.WH_Step_Price2=get(handles.WH_Step_Price_Text2,'String');
    guidata(hObject, handles);

function WH_Step_Price_Text2_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text2,'String','');

function WH_Step_Price_Text2_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text2,'String','');

function WH_Step_Price_Text2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Price_Text3_Callback(hObject, eventdata, handles)

    handles.WH_Step_Price3=get(handles.WH_Step_Price_Text3,'String');
    guidata(hObject, handles);

function WH_Step_Price_Text3_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text3,'String','');

function WH_Step_Price_Text3_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text3,'String','');

function WH_Step_Price_Text3_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Price_Text4_Callback(hObject, eventdata, handles)

    handles.WH_Step_Price4=get(handles.WH_Step_Price_Text4,'String');
    guidata(hObject, handles);

function WH_Step_Price_Text4_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text4,'String','');

function WH_Step_Price_Text4_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text4,'String','');
```

```
function WH_Step_Price_Text4_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function WH_Step_Price_Text5_Callback(hObject, eventdata, handles)

    handles.WH_Step_Price5=get(handles.WH_Step_Price_Text5,'String');
    guidata(hObject, handles);

function WH_Step_Price_Text5_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text5,'String','');

function WH_Step_Price_Text5_KeyPressFcn(hObject, eventdata, handles)

    set(handles.WH_Step_Price_Text5,'String','');

function WH_Step_Price_Text5_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function Named_Price_Text1_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Named_Price_Text1,'String','');

function Named_Price_Text1_KeyPressFcn(hObject, eventdata, handles)

    set(handles.Named_Price_Text1,'String','');

function Named_Price_Text2_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Named_Price_Text2,'String','');

function Named_Price_Text2_KeyPressFcn(hObject, eventdata, handles)

    set(handles.Named_Price_Text2,'String','');

function Named_Price_Text3_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Named_Price_Text3,'String','');

function Named_Price_Text3_KeyPressFcn(hObject, eventdata, handles)

    set(handles.Named_Price_Text3,'String','');

function Named_Price_Text4_ButtonDownFcn...
        (hObject, eventdata, handles)
```

```
    set(handles.Named_Price_Text4,'String','');

function Named_Price_Text4_KeyPressFcn(hObject, eventdata, handles)

    set(handles.Named_Price_Text4,'String','');

function Named_Price_Text5_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Named_Price_Text5,'String','');

function Named_Price_Text5_KeyPressFcn(hObject, eventdata, handles)

    set(handles.Named_Price_Text5,'String','');

function Named_Price_Text6_ButtonDownFcn...
        (hObject, eventdata, handles)

    set(handles.Named_Price_Text6,'String','');

function Named_Price_Text6_KeyPressFcn(hObject, eventdata, handles)

    set(handles.Named_Price_Text6,'String','');
```

APPENDIX C

MATLAB USER INTERFACE SOUCE CODES – PRICE_NAMING_ASSISTANT

```
function varargout = Price_Naming_Assistant(varargin)


gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Price_Naming_Assistant_OpeningFcn, ...
                   'gui_OutputFcn',  @Price_Naming_Assistant_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function Price_Naming_Assistant_OpeningFcn...
    (hObject, eventdata, handles, varargin)

hold off

    handles.Hour_Deadline_Assistant=get...
        (handles.Hour_Deadline_Assistant_Text,'String');
    handles.AMPM_Deadline_Assistant=get...
        (handles.AMPM_Deadline_Assistant_Text,'String');

    string_mins_deadline=get(handles.Minute_Deadline_Assistant_Text,'String');
    num_mins_deadline=str2double(string_mins_deadline);

    if and((num_mins_deadline<15),(num_mins_deadline>=0)),
        handles.Minute_Deadline_Assistant='00';
    end
    if and((num_mins_deadline<30),(num_mins_deadline>=15)),
        handles.Minute_Deadline_Assistant='15';
    end
    if and((num_mins_deadline<45),(num_mins_deadline>=30)),
        handles.Minute_Deadline_Assistant='30';
    end
    if and((num_mins_deadline<=59),(num_mins_deadline>=45)),
        handles.Minute_Deadline_Assistant='45';
    end

    if isempty(handles.Hour_Deadline_Assistant)==0,
        if isempty(handles.Minute_Deadline_Assistant)==0,
            if isempty(handles.AMPM_Deadline_Assistant)==0,

    handles.Deadline_time_Assistant=[handles.Hour_Deadline_Assistant,...
        ':',...
        handles.Minute_Deadline_Assistant,...
        ' ',...
        handles.AMPM_Deadline_Assistant];

    handles.location_Deadline=[];

    if size(handles.Deadline_time_Assistant,2)==8,
```

```
  for index=1:size(handles.shifted_all_day_time_frame,1)

    if handles.shifted_all_day_time_frame(index,1)==...
            handles.Deadline_time_Assistant(1);
    if handles.shifted_all_day_time_frame(index,2)==...
            handles.Deadline_time_Assistant(2);
    if handles.shifted_all_day_time_frame(index,4)==...
            handles.Deadline_time_Assistant(4);
    if handles.shifted_all_day_time_frame(index,5)==...
            handles.Deadline_time_Assistant(5);
    if handles.shifted_all_day_time_frame(index,7)==...
            handles.Deadline_time_Assistant(7);
        handles.location_Deadline=index;
    end
    end
    end
    end
    end

  end

end

if size(handles.Deadline_time_Assistant,2)==7,
  for index=1:size(handles.shifted_all_day_time_frame,1)

    if handles.shifted_all_day_time_frame(index,2)==...
            handles.Deadline_time_Assistant(1);
    if handles.shifted_all_day_time_frame(index,4)==...
            handles.Deadline_time_Assistant(3);
    if handles.shifted_all_day_time_frame(index,5)==...
            handles.Deadline_time_Assistant(4);
    if handles.shifted_all_day_time_frame(index,7)==...
            handles.Deadline_time_Assistant(6);
        if handles.shifted_all_day_time_frame(index,1)==' ';
        handles.location_Deadline=index;
        end
    end
    end
    end
    end

  end

end

        if handles.location_Deadline==[],

            handles.location_Deadline=[];
        end

    handles.deadline_price=round...
        (handles.shifted_robusted_smoothed_north_price_4_days...
        (handles.location_Deadline));

  plot(handles.shifted_robusted_smoothed_north_price_4_days,'g*-')
  hold on
  plot(handles.current_north_zone_prices,'-.or')
  ylim([0 120])
```

```
      set(gca,'XTick',1:8:handles.num_all_day_time_frame)
      set(gca,'XTickLabel',handles.shifted_all_day_time_frame_every_2hours)
      set(gca,'FontSize',8)
      grid on
      legend('Estimated Prices','Current Price');

        hold on
        handles.h_figure_deadline=plot...
            (handles.location_Deadline,0:120,'rx:');
        handles.shifted_price_assistant=...
            handles.shifted_robusted_smoothed_north_price_4_days...
            (1:handles.location_Deadline);

handles.min_shifted_price_assistant=min(handles.shifted_price_assistant);
        handles.location_min_shifted_price_assistant=...
            find(handles.shifted_price_assistant==...
            handles.min_shifted_price_assistant);
        handles.location_min_shifted_price_assistant=...
            handles.location_min_shifted_price_assistant(1);

        handles.min_shifted_time_assistant=...
            handles.shifted_all_day_time_frame...
            (handles.location_min_shifted_price_assistant,:);
        hold on
        handles.h_figure_min=plot...
            (handles.location_min_shifted_price_assistant,0:120,'cx:');

    handles.datenum_min_shifted_time_assistant=datenum...
        (handles.min_shifted_time_assistant);
    handles.datevec_min_shifted_time_assistant=datevec...
        (handles.datenum_min_shifted_time_assistant);

    handles.datevec_now=fix(datevec(now));

% match elements 2 and 3 (month and day) to current date
% same date
    if handles.datevec_now(4)<=...
            handles.datevec_min_shifted_time_assistant(4),
        handles.datevec_min_shifted_time_assistant(2:3)=...
            handles.datevec_now(2:3);
    end
% different date (same month or next month is all right)
    if handles.datevec_now(4)>...
            handles.datevec_min_shifted_time_assistant(4),
        handles.datevec_min_shifted_time_assistant(2)=...
            handles.datevec_now(2);
        handles.datevec_min_shifted_time_assistant(3)=...
            handles.datevec_now(3)+1;
    end

    handles.new_datenum_min_shifted_time_assistant=datenum...
        (handles.datevec_min_shifted_time_assistant);

    % reach estimate min price time
    if handles.new_datenum_min_shifted_time_assistant<=now,


        h = actxserver('X10.ActiveHome.1');
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
```

```
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    h = actxserver('X10.ActiveHome.1');

    handles.X10_address7=get(handles.X10_Address_Text7,'String');
    handles.Named_Price7=str2double(get...
        (handles.Named_Price_Text7,'String'));

    if  handles.current_north_zone_prices <= ...
            handles.min_shifted_price_assistant,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    if  handles.current_north_zone_prices <= handles.Named_Price7,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    handles.datenum_Deadline_time_Assistant=datenum...
        (handles.Deadline_time_Assistant);
    handles.datevec_Deadline_time_Assistant=datevec...
        (handles.datenum_Deadline_time_Assistant);

    handles.datevec_now=fix(datevec(now));

% match elements 2 and 3 (month and day) to current date
% same date
    if handles.datevec_now(4)<=handles.datevec_Deadline_time_Assistant(4),
        handles.datevec_Deadline_time_Assistant(2:3)=...
            handles.datevec_now(2:3);
    end
% different date (same month or next month is all right)
    if handles.datevec_now(4)>...
            handles.datevec_Deadline_time_Assistant(4),
        handles.datevec_Deadline_time_Assistant(2)=...
            handles.datevec_now(2);
        handles.datevec_Deadline_time_Assistant(3)=...
            handles.datevec_now(3)+1;
    end

    handles.new_datenum_Deadline_time_Assistant=datenum...
        (handles.datevec_Deadline_time_Assistant);

    % reach min estimated price

    % reach deadline
    if handles.new_datenum_Deadline_time_Assistant<=now,


        h = actxserver('X10.ActiveHome.1');
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end
```

```
    set(handles.Estimated_Hour_Assistant_Text,'String', ...
        handles.min_shifted_time_assistant(1:2));
    set(handles.Estimated_Minute_Assistant_Text,'String', ...
        handles.min_shifted_time_assistant(4:5));
    set(handles.Estimated_AMPM_Assistant_Text,'String', ...
        handles.min_shifted_time_assistant(7:8));
    set(handles.Estimated_Price_Assistant_Text,'String', ...
        num2str(round(handles.min_shifted_price_assistant)));

            end
        end
    end

        [north_zone_prices,num_location_time_frame,new_time_frame]...
            =ERCOT_prices;

        length_north_zone_prices=length(north_zone_prices);
        handles.length_north_zone_prices=length_north_zone_prices;
        current_north_zone_prices=north_zone_prices...
            (length_north_zone_prices);

        handles.current_north_zone_prices=current_north_zone_prices;
        north_price_date_values;
        north_price_4_days=zeros(96,4);
        north_price_4_days(:,1)=north_price_date_1;
        north_price_4_days(:,2)=north_price_date_2;
        north_price_4_days(:,3)=north_price_date_3;
        north_price_4_days(:,4)=north_price_date_4;
        north_price_4_days;

        average_north_price_4_days=mean(north_price_4_days,2);
%       smoothed_average_north_price_4_days=smooth...
%       (average_north_price_4_days);

        robusted_smoothed_north_price_4_days=smooth...
            (average_north_price_4_days,0.1,'rloess');

        shifted_robusted_smoothed_north_price_4_days = ...
        circshift(robusted_smoothed_north_price_4_days,-...
        (length_north_zone_prices-1));

        handles.shifted_robusted_smoothed_north_price_4_days=...
            shifted_robusted_smoothed_north_price_4_days;


        handles.all_day_time_frame=datestr(xlsread...
            ('2008-03_BES', '01', 'b6:b101'));
        handles.num_all_day_time_frame=size(handles.all_day_time_frame,1);

        handles.shifted_all_day_time_frame=circshift...
            (handles.all_day_time_frame,-...
            (handles.length_north_zone_prices-1));
        handles.shifted_all_day_time_frame_every_2hours=...
            handles.shifted_all_day_time_frame...
            (1:8:handles.num_all_day_time_frame,:);

        handles.shifted_all_day_time_frame=...
            handles.shifted_all_day_time_frame;
```

```
      plot(handles.shifted_robusted_smoothed_north_price_4_days,'g*-')
      hold on
      plot(handles.current_north_zone_prices,'-.or')
      ylim([0 120])
      set(gca,'XTick',1:8:handles.num_all_day_time_frame)
      set(gca,'XTickLabel',...
          handles.shifted_all_day_time_frame_every_2hours)
      set(gca,'FontSize',8)
      grid on
      legend('Estimated Prices','Current Price');

    handles.north_zone_prices=north_zone_prices;
    handles.current_north_zone_prices=current_north_zone_prices;

    set(handles.Current_Price_Text,'String', ...
        num2str(handles.current_north_zone_prices));

    % get string time in HH:MM AM or PM
    current_time_string=datestr(now,16);
    handles.current_time_string=current_time_string;
    set(handles.Current_Hour_Text,'String', ...
        handles.current_time_string(1:2));
    set(handles.Current_Minute_Text,'String', ...
        handles.current_time_string(4:5));
    set(handles.Current_AMPM_Text,'String', ...
        handles.current_time_string(7:8));

% num = xlsread('2008-03_BES', '04', 'c5:c101')

    handles.update_Price_Naming_Assistant=60*str2double...
        (get(handles.update_Price_Naming_Assistant_Text,'String'));

    handles.X10_address7=get(handles.X10_Address_Text7,'String');

    h = actxserver('X10.ActiveHome.1');

    handles.on_status7=SendAction(h, 'queryplc', ...
        [handles.X10_address7 ' on']);
    if handles.on_status7==1,
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    else
        set(handles.Turn_On_Button7,'Value',0);
        set(handles.Turn_Off_Button7,'Value',1);
    end

    handles.Named_Price7=str2double(get(...
        handles.Named_Price_Text7,'String'));

    if  handles.current_north_zone_prices <= handles.Named_Price7,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

handles.output = hObject;

guidata(hObject, handles);
```

```
function varargout = Price_Naming_Assistant_OutputFcn...
    (hObject, eventdata, handles)

varargout{1} = handles.output;

function X10_Address_Text7_Callback(hObject, eventdata, handles)

    handles.X10_address7=get(handles.X10_Address_Text7,'String');

function X10_Address_Text7_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Named_Price_Text7_Callback(hObject, eventdata, handles)

    handles.Named_Price7=str2double(get(...
        handles.Named_Price_Text7,'String'));

function Named_Price_Text7_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Reset_Timer_Assistant_Callback(hObject, eventdata, handles)

delete (handles.t_Assistant);

function update_Price_Naming_Assistant_Text_Callback...
    (hObject, eventdata, handles)

    handles.update_Price_Naming_Assistant=60*str2double...
        (get(handles.update_Price_Naming_Assistant_Text,'String'));

    handles.t_Assistant = timer('Period', ...
        handles.update_Price_Naming_Assistant, ...
        'ExecutionMode','fixedSpacing', ...
        'StartFcn', {'Price_Naming_Assistant'}, ...
        'TimerFcn', {'Price_Naming_Assistant'});

    start(handles.t_Assistant);
    guidata(hObject, handles);

function update_Price_Naming_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Hour_Deadline_Assistant_Text_Callback...
    (hObject, eventdata, handles)
```

```
    handles.Hour_Deadline_Assistant=get...
        (handles.Hour_Deadline_Assistant_Text,'String');
    handles.AMPM_Deadline_Assistant=get...
        (handles.AMPM_Deadline_Assistant_Text,'String');

    string_mins_deadline=get...
        (handles.Minute_Deadline_Assistant_Text,'String');
    num_mins_deadline=str2double(string_mins_deadline);

    if and((num_mins_deadline<15),(num_mins_deadline>=0)),
        handles.Minute_Deadline_Assistant=0;
    end
    if and((num_mins_deadline<30),(num_mins_deadline>=15)),
        handles.Minute_Deadline_Assistant=15;
    end
    if and((num_mins_deadline<45),(num_mins_deadline>=30)),
        handles.Minute_Deadline_Assistant=30;
    end
    if and((num_mins_deadline<=59),(num_mins_deadline>=45)),
        handles.Minute_Deadline_Assistant=0;
    end

function Hour_Deadline_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Minute_Deadline_Assistant_Text_Callback...
    (hObject, eventdata, handles)

    handles.Hour_Deadline_Assistant=get...
        (handles.Hour_Deadline_Assistant_Text,'String');
    handles.AMPM_Deadline_Assistant=get...
        (handles.AMPM_Deadline_Assistant_Text,'String');

    string_mins_deadline=get...
        (handles.Minute_Deadline_Assistant_Text,'String');
    num_mins_deadline=str2double(string_mins_deadline);

    if and((num_mins_deadline<15),(num_mins_deadline>=0)),
        handles.Minute_Deadline_Assistant='00';
    end
    if and((num_mins_deadline<30),(num_mins_deadline>=15)),
        handles.Minute_Deadline_Assistant='15';
    end
    if and((num_mins_deadline<45),(num_mins_deadline>=30)),
        handles.Minute_Deadline_Assistant='30';
    end
    if and((num_mins_deadline<=59),(num_mins_deadline>=45)),
        handles.Minute_Deadline_Assistant='45';
    end

function Minute_Deadline_Assistant_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end

function AMPM_Deadline_Assistant_Text_Callback(hObject, eventdata, handles)

    handles.Hour_Deadline_Assistant=get...
        (handles.Hour_Deadline_Assistant_Text,'String');
    handles.AMPM_Deadline_Assistant=get...
        (handles.AMPM_Deadline_Assistant_Text,'String');

    string_mins_deadline=get...
        (handles.Minute_Deadline_Assistant_Text,'String');
    num_mins_deadline=str2double(string_mins_deadline);

    if and((num_mins_deadline<15),(num_mins_deadline>=0)),
        handles.Minute_Deadline_Assistant='00';
    end
    if and((num_mins_deadline<30),(num_mins_deadline>=15)),
        handles.Minute_Deadline_Assistant='15';
    end
    if and((num_mins_deadline<45),(num_mins_deadline>=30)),
        handles.Minute_Deadline_Assistant='30';
    end
    if and((num_mins_deadline<=59),(num_mins_deadline>=45)),
        handles.Minute_Deadline_Assistant='45';
    end

function AMPM_Deadline_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Set_Deadline_Assistant_Callback...
    (hObject, eventdata, handles)

    handles.Hour_Deadline_Assistant=get...
        (handles.Hour_Deadline_Assistant_Text,'String');
    handles.AMPM_Deadline_Assistant=get...
        (handles.AMPM_Deadline_Assistant_Text,'String');

    string_mins_deadline=get...
        (handles.Minute_Deadline_Assistant_Text,'String');
    num_mins_deadline=str2double(string_mins_deadline);

    if and((num_mins_deadline<15),(num_mins_deadline>=0)),
        handles.Minute_Deadline_Assistant='00';
    end
    if and((num_mins_deadline<30),(num_mins_deadline>=15)),
        handles.Minute_Deadline_Assistant='15';
    end
    if and((num_mins_deadline<45),(num_mins_deadline>=30)),
        handles.Minute_Deadline_Assistant='30';
    end
    if and((num_mins_deadline<=59),(num_mins_deadline>=45)),
        handles.Minute_Deadline_Assistant='45';
    end
```

```
handles.Deadline_time_Assistant=[handles.Hour_Deadline_Assistant,...
    ':',...
    handles.Minute_Deadline_Assistant,...
    ' ',...
    handles.AMPM_Deadline_Assistant];

handles.location_Deadline=[];


if size(handles.Deadline_time_Assistant,2)==8,
  for index=1:size(handles.shifted_all_day_time_frame,1)

    if handles.shifted_all_day_time_frame(index,1)==...
            handles.Deadline_time_Assistant(1);
    if handles.shifted_all_day_time_frame(index,2)==...
            handles.Deadline_time_Assistant(2);
    if handles.shifted_all_day_time_frame(index,4)==...
            handles.Deadline_time_Assistant(4);
    if handles.shifted_all_day_time_frame(index,5)==...
            handles.Deadline_time_Assistant(5);
    if handles.shifted_all_day_time_frame(index,7)==...
            handles.Deadline_time_Assistant(7);
        handles.location_Deadline=index;
    end
    end
    end
    end
    end

  end

end

if size(handles.Deadline_time_Assistant,2)==7,
  for index=1:size(handles.shifted_all_day_time_frame,1)

    if handles.shifted_all_day_time_frame(index,2)==...
            handles.Deadline_time_Assistant(1);
    if handles.shifted_all_day_time_frame(index,4)==...
            handles.Deadline_time_Assistant(3);
    if handles.shifted_all_day_time_frame(index,5)==...
            handles.Deadline_time_Assistant(4);
    if handles.shifted_all_day_time_frame(index,7)==...
            handles.Deadline_time_Assistant(6);
        if handles.shifted_all_day_time_frame(index,1)==' ';
        handles.location_Deadline=index;
        end
    end
    end
    end
    end

  end

end

        if handles.location_Deadline==[],
```

```
                handles.location_Deadline=[];
            end

        handles.deadline_price=round...
            (handles.shifted_robusted_smoothed_north_price_4_days...
            (handles.location_Deadline));

hold off

plot(handles.shifted_robusted_smoothed_north_price_4_days,'g*-')
      hold on
      plot(handles.current_north_zone_prices,'-.or')
      ylim([0 120])
      set(gca,'XTick',1:8:handles.num_all_day_time_frame)
      set(gca,'XTickLabel',...
          handles.shifted_all_day_time_frame_every_2hours)
      set(gca,'FontSize',8)
      grid on
      legend('Estimated Prices','Current Price');

        hold on
        handles.h_figure_deadline=plot...
            (handles.location_Deadline,0:120,'rx:');

        handles.shifted_price_assistant=...
            handles.shifted_robusted_smoothed_north_price_4_days...
            (1:handles.location_Deadline);
        handles.min_shifted_price_assistant=...
            min(handles.shifted_price_assistant);
        handles.location_min_shifted_price_assistant=...
            find(handles.shifted_price_assistant==...
            handles.min_shifted_price_assistant);
        handles.location_min_shifted_price_assistant=...
            handles.location_min_shifted_price_assistant(1);

        handles.min_shifted_time_assistant=...
            handles.shifted_all_day_time_frame...
            (handles.location_min_shifted_price_assistant,:);
        hold on
        handles.h_figure_min=plot...
            (handles.location_min_shifted_price_assistant,0:120,'cx:');

    handles.datenum_min_shifted_time_assistant=datenum...
        (handles.min_shifted_time_assistant);
    handles.datevec_min_shifted_time_assistant=datevec...
        (handles.datenum_min_shifted_time_assistant);

    handles.datevec_now=fix(datevec(now));

% match elements 2 and 3 (month and day) to current date
% same date
    if handles.datevec_now(4)<=...
            handles.datevec_min_shifted_time_assistant(4),
        handles.datevec_min_shifted_time_assistant(2:3)=...
            handles.datevec_now(2:3);
    end
% different date (same month or next month is all right)
    if handles.datevec_now(4)>...
            handles.datevec_min_shifted_time_assistant(4),
```

```
        handles.datevec_min_shifted_time_assistant(2)=...
            handles.datevec_now(2);
        handles.datevec_min_shifted_time_assistant(3)=...
            handles.datevec_now(3)+1;
    end

    handles.new_datenum_min_shifted_time_assistant=datenum...
        (handles.datevec_min_shifted_time_assistant);

    % reach estimate min price time
    if handles.new_datenum_min_shifted_time_assistant<=now,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    h = actxserver('X10.ActiveHome.1');

    handles.X10_address7=get(handles.X10_Address_Text7,'String');
    handles.Named_Price7=str2double(get...
        (handles.Named_Price_Text7,'String'));

    if  handles.current_north_zone_prices <= ...
            handles.min_shifted_price_assistant,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    if  handles.current_north_zone_prices <= handles.Named_Price7,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    handles.datenum_Deadline_time_Assistant=datenum...
        (handles.Deadline_time_Assistant);
    handles.datevec_Deadline_time_Assistant=datevec...
        (handles.datenum_Deadline_time_Assistant);

    handles.datevec_now=fix(datevec(now));

% match elements 2 and 3 (month and day) to current date
% same date
    if handles.datevec_now(4)<=...
            handles.datevec_Deadline_time_Assistant(4),
        handles.datevec_Deadline_time_Assistant(2:3)=...
            handles.datevec_now(2:3);
    end
% different date (same month or next month is all right)
    if handles.datevec_now(4)>...
            handles.datevec_Deadline_time_Assistant(4),
        handles.datevec_Deadline_time_Assistant(2)=...
            handles.datevec_now(2);
        handles.datevec_Deadline_time_Assistant(3)=...
            handles.datevec_now(3)+1;
    end

    handles.new_datenum_Deadline_time_Assistant=datenum...
```

```
        (handles.datevec_Deadline_time_Assistant);

    % reach deadline
    if handles.new_datenum_Deadline_time_Assistant<=now,
        SendAction(h, 'sendplc', [handles.X10_address7 ' on']);
        set(handles.Turn_On_Button7,'Value',1);
        set(handles.Turn_Off_Button7,'Value',0);
    end

    set(handles.Estimated_Hour_Assistant_Text,'String', ...
        handles.min_shifted_time_assistant(1:2));
    set(handles.Estimated_Minute_Assistant_Text,'String', ...
        handles.min_shifted_time_assistant(4:5));
    set(handles.Estimated_AMPM_Assistant_Text,'String', ...
        handles.min_shifted_time_assistant(7:8));
    set(handles.Estimated_Price_Assistant_Text,'String', ...
        num2str(round(handles.min_shifted_price_assistant)));

function Turn_On_Button7_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');

    handles.X10_address7=get(handles.X10_Address_Text7,'String');
    SendAction(h, 'sendplc', [handles.X10_address7 ' on']);

function Turn_Off_Button7_Callback(hObject, eventdata, handles)

    h = actxserver('X10.ActiveHome.1');
    handles.X10_address7=get(handles.X10_Address_Text7,'String');
    SendAction(h, 'sendplc', [handles.X10_address7 ' off']);

function Home_Callback(hObject, eventdata, handles)

Home

function X10_PLC_RF_Callback(hObject, eventdata, handles)

X10_PLC_RF

function Price_Naming_Assistant_Callback(hObject, eventdata, handles)

Price_Naming_Assistant

function Real_Time_Price_Callback(hObject, eventdata, handles)
Real_Time_Price

function Real_Time_Outdoor_Temperature_Callback...
    (hObject, eventdata, handles
Real_Time_Outdoor_Temperature

function Surveillance_Camera_Callback(hObject, eventdata, handles)
Surveillance_Camera

function Current_Hour_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Current_Minute_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_AMPM_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Estimated_Hour_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Estimated_Minute_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Estimated_AMPM_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Estimated_Price_Assistant_Text_Callback...
    (hObject, eventdata, handles)

function Estimated_Price_Assistant_Text_CreateFcn...
    (hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Price_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function X10_Address_Text7_KeyPressFcn(hObject, eventdata, handles)
```

```
function Named_Price_Text7_KeyPressFcn(hObject, eventdata, handles)

function update_Price_Naming_Assistant_Text_KeyPressFcn...
    (hObject, eventdata, handles)
```

APPENDIX D

MATLAB USER INTERFACE SOUCE CODES – REAL_TIME_PRICE

```
function varargout = Real_Time_Price(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Real_Time_Price_OpeningFcn, ...
                   'gui_OutputFcn',  @Real_Time_Price_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function Real_Time_Price_OpeningFcn(hObject, ...
    eventdata, handles, varargin)
        [north_zone_prices,num_location_time_frame,...
            new_time_frame]=ERCOT_prices;
        % show x tick only one every 2 hours
        plot(north_zone_prices,'-.or')
        set(gca,'XTick',1:8:num_location_time_frame)
        new_time_frame_every_2hours=new_time_frame...
            (1:8:num_location_time_frame,:);
        new_time_frame_every_2hours;

        set(gca,'XTickLabel',new_time_frame_every_2hours)
        set(gca,'FontSize',8)
        grid on

        length_north_zone_prices=length(north_zone_prices);
        current_north_zone_prices=north_zone_prices...
            (length_north_zone_prices);

    handles.north_zone_prices=north_zone_prices;
    handles.current_north_zone_prices=current_north_zone_prices;

    set(handles.Current_Price_Text,'String', num2str...
        (handles.current_north_zone_prices));

    % get string time in HH:MM AM or PM
    current_time_string=datestr(now,16);
    handles.current_time_string=current_time_string;
    set(handles.Current_Hour_Text,'String', ...
        handles.current_time_string(1:2));
    set(handles.Current_Minute_Text,'String', ...
        handles.current_time_string(4:5));
    set(handles.Current_AMPM_Text,'String', ...
        handles.current_time_string(7:8));

handles.output = hObject;
guidata(hObject, handles);

function varargout = Real_Time_Price_OutputFcn...
    (hObject, eventdata, handles)
varargout{1} = handles.output;
```

```
function pushbutton1_Callback(hObject, eventdata, handles)
        [north_zone_prices,num_location_time_frame,...
            new_time_frame]=ERCOT_prices;
        % show x tick only one every 2 hours
        plot(north_zone_prices,'-.or')
        set(gca,'XTick',1:8:num_location_time_frame)
        new_time_frame_every_2hours=new_time_frame...
            (1:8:num_location_time_frame,:);
        new_time_frame_every_2hours;

        set(gca,'XTickLabel',new_time_frame_every_2hours)
        set(gca,'FontSize',8)
        grid on

        length_north_zone_prices=length(north_zone_prices);
        current_north_zone_prices=north_zone_prices...
            (length_north_zone_prices);

    handles.north_zone_prices=north_zone_prices;
    handles.current_north_zone_prices=current_north_zone_prices;

    set(handles.Current_Price_Text,'String', num2str...
        (handles.current_north_zone_prices));

    % get string time in HH:MM AM or PM
    current_time_string=datestr(now,16);
    handles.current_time_string=current_time_string;
    set(handles.Current_Hour_Text,'String', ...
        handles.current_time_string(1:2));
    set(handles.Current_Minute_Text,'String', ...
        handles.current_time_string(4:5));
    set(handles.Current_AMPM_Text,'String', ...
        handles.current_time_string(7:8));

handles.output = hObject;

guidata(hObject, handles);

function Current_Price_Text_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Hour_Text_CreateFcn(hObject, ...
    eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Minute_Text_Callback(hObject, eventdata, handles)

function Current_Minute_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end

function Current_AMPM_Text_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Home_Callback(hObject, eventdata, handles)
Home

function X10_PLC_RF_Callback(hObject, eventdata, handles)
X10_PLC_RF

function Price_Naming_Assistant_Callback(hObject, ...
    eventdata, handles)
Price_Naming_Assistant

function Real_Time_Price_Callback(hObject, eventdata, handles)
Real_Time_Price

function Real_Time_Outdoor_Temperature_Callback...
    (hObject, eventdata, handles)
Real_Time_Outdoor_Temperature

function Surveillance_Camera_Callback(hObject, eventdata, handles)
Surveillance_Camera
```

APPENDIX E

MATLAB USER INTERFACE SOUCE CODES – REAL_TIME_OUTDOOR_TEMPERATURE

```
function varargout = Real_Time_Outdoor_Temperature(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', ...
                   @Real_Time_Outdoor_Temperature_OpeningFcn, ...
                   'gui_OutputFcn',  ...
                   @Real_Time_Outdoor_Temperature_OutputFcn, ...
                   'gui_LayoutFcn',  [], ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function Real_Time_Outdoor_Temperature_OpeningFcn...
    (hObject, eventdata, handles, varargin)
        % plot weather forecast
        [Outdoor_Temperature, new_time_frame]=...
            outdoor_weather_from_findlocalweather;
        % show x tick only one every 4 hours
        plot(Outdoor_Temperature,'--*')
        set(gca,'XTick',1:4:48)

        new_time_frame_every_4hours=new_time_frame(1:4:48,:);
        new_time_frame_every_4hours;

        set(gca,'XTickLabel',new_time_frame_every_4hours)
        set(gca,'FontSize',8)

        grid on

        length_Outdoor_Temperature=length(Outdoor_Temperature);

        % pick the first element as the current temperature
        current_Outdoor_Temperature=Outdoor_Temperature(1);

    handles.Outdoor_Temperature=Outdoor_Temperature;
    handles.current_Outdoor_Temperature=...
        current_Outdoor_Temperature;

    set(handles.current_Outdoor_Temperature_Text,'String',...
        num2str(handles.current_Outdoor_Temperature));

    % get string time in HH:MM AM or PM
    current_time_string=datestr(now,16);
    handles.current_time_string=current_time_string;
    set(handles.Current_Hour_Text,'String', ...
        handles.current_time_string(1:2));
    set(handles.Current_Minute_Text,'String', ...
        handles.current_time_string(4:5));
    set(handles.Current_AMPM_Text,'String', ...
        handles.current_time_string(7:8));
```

```
% Choose default command line output for
% Real_Time_Outdoor_Temperature
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

function varargout = Real_Time_Outdoor_Temperature_OutputFcn...
    (hObject, eventdata, handles)
varargout{1} = handles.output;

function current_Outdoor_Temperature_Text_CreateFcn...
    (hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Hour_Text_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_Minute_Text_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Current_AMPM_Text_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbutton1_Callback(hObject, eventdata, handles)
        % plot weather forecast
        [Outdoor_Temperature, new_time_frame]=...
            outdoor_weather_from_findlocalweather;
        % show x tick only one every 4 hours
        plot(Outdoor_Temperature,'--*')
        set(gca,'XTick',1:4:48)

        new_time_frame_every_4hours=new_time_frame(1:4:48,:);
        new_time_frame_every_4hours;

        set(gca,'XTickLabel',new_time_frame_every_4hours)
        set(gca,'FontSize',8)

        grid on

        length_Outdoor_Temperature=length(Outdoor_Temperature);

        % pick the first element as the current temperature
        current_Outdoor_Temperature=Outdoor_Temperature(1);

    handles.Outdoor_Temperature=Outdoor_Temperature;
    handles.current_Outdoor_Temperature=...
        current_Outdoor_Temperature;
```

```
    set(handles.current_Outdoor_Temperature_Text,'String',...
        num2str(handles.current_Outdoor_Temperature));

    % get string time in HH:MM AM or PM
    current_time_string=datestr(now,16);
    handles.current_time_string=current_time_string;
    set(handles.Current_Hour_Text,'String', ...
        handles.current_time_string(1:2));
    set(handles.Current_Minute_Text,'String', ...
        handles.current_time_string(4:5));
    set(handles.Current_AMPM_Text,'String', ...
        handles.current_time_string(7:8));

function Home_Callback(hObject, eventdata, handles)
Home

function X10_PLC_RF_Callback(hObject, eventdata, handles)
X10_PLC_RF

function Price_Naming_Assistant_Callback...
    (hObject, eventdata, handles)
Price_Naming_Assistant

function Real_Time_Price_Callback(hObject, eventdata, handles)
Real_Time_Price

function Real_Time_Outdoor_Temperature_Callback...
    (hObject, eventdata, handles)
Real_Time_Outdoor_Temperature

function Surveillance_Camera_Callback(hObject, eventdata, handles)
Surveillance_Camera
```

APPENDIX F

MATLAB USER INTERFACE SOUCE CODES – SURVEILLANCE_CAMERA

```
function varargout = Surveillance_Camera(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', ...
                   @Surveillance_Camera_OpeningFcn, ...
                   'gui_OutputFcn',  ...
                   @Surveillance_Camera_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function Surveillance_Camera_OpeningFcn(hObject, ...
    eventdata, handles, varargin)
        % Create a video input object.
        handles.vid = videoinput('winvideo');
        % Create an image object for previewing.
        vidRes = get(handles.vid, 'VideoResolution');
        nBands = get(handles.vid, 'NumberOfBands');
        hImage = image( zeros(vidRes(2), vidRes(1), nBands) );
        preview(handles.vid, hImage);

% Choose default command line output for Surveillance_Camera
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

function varargout = Surveillance_Camera_OutputFcn...
    (hObject, eventdata, handles)
varargout{1} = handles.output;

% --- Executes on button press in Stop.
function Stop_Callback(hObject, eventdata, handles)
stoppreview(handles.vid)

% --- Executes on button press in Resume.
function Resume_Callback(hObject, eventdata, handles)
        % Create a video input object.
        handles.vid = videoinput('winvideo');
        % Create an image object for previewing.
        vidRes = get(handles.vid, 'VideoResolution');
        nBands = get(handles.vid, 'NumberOfBands');
        hImage = image( zeros(vidRes(2), vidRes(1), nBands) );
        preview(handles.vid, hImage);

handles.output = hObject;
guidata(hObject, handles);

function Home_Callback(hObject, eventdata, handles)
Home
```

```
function X10_PLC_RF_Callback(hObject, eventdata, handles)
X10_PLC_RF_Callback

function Price_Naming_Assistant_Callback...
    (hObject, eventdata, handles)
Price_Naming_Assistant

function Real_Time_Price_Callback(hObject, eventdata, handles)
Real_Time_Price

function Real_Time_Outdoor_Temperature_Callback...
    (hObject, eventdata, handles)
Real_Time_Outdoor_Temperature

function Surveillance_Camera_Callback...
    (hObject, eventdata, handles)
Surveillance_Camera
```

APPENDIX G

MATLAB USER INTERFACE SOUCE CODES – ERCOT_PRICES

```
function [north_zone_prices,num_location_time_frame,new_time_frame]...
    =ERCOT_prices,
s = urlread...
    ('http://mospublic.ercot.com/ercot/jsp/balancing_services_mcp.jsp');
location=findstr(s,'$');
num_location=length(location);
all_prices=zeros(1,num_location);

for index=1:num_location,
    % when the price is single digit, the 5th character is either ...
    % negative number or positive
    if (s(location(index)+5)==')')|(s(location(index)+5)=='&')
        price=s(location(index)+1:location(index)+4);
    else
        price=s(location(index)+1:location(index)+5);
    end

    temp=str2num(price);
    all_prices(index)=temp;
end

all_prices;
price_row=num_location/4;
north_zone_prices=zeros(1,price_row);

for index2=1:price_row,
north_zone_prices(index2)=all_prices((index2*4)-2);
end

north_zone_prices;
location_for_time_frame=findstr(s,' </FONT></TD>');
num_location_for_time_frame=length(location_for_time_frame);

location_time_frame=location_for_time_frame(1:5:num_location_for_time_frame);
num_location_time_frame=length(location_time_frame);

time_frame='';

for index3=1:num_location_time_frame,
temp=time_frame;
time_frame=s(location_time_frame(index3)-4:location_time_frame(index3)-1);
time_frame=strvcat(temp,time_frame);
end

time_frame;

% arrange in the MATLAB format
HH=time_frame(:,1:2);
MM=time_frame(:,3:4);

% insert ':' in every point of time
HH(:,3)=':';
HH(:,4:5)=MM;
new_time_frame=HH;

% delete 0 in front of single digit hour
time_series_hour=datenum(new_time_frame);
% style 16
new_time_frame=datestr(time_series_hour,16);
```

APPENDIX H

MATLAB USER INTERFACE SOUCE CODES – OUTDOOR_WEATHER_FROM_FINDLOCALWEATHER

```
function [Temperature, new_time_frame]=...
    outdoor_weather_from_findlocalweather,

% next 48 hour data
temperature_web_text1 = urlread...
    ('http://www.findlocalweather.com/by_the_hour/tx/arlington.html');
location=findstr(temperature_web_text1,'&deg;F');
num_location=length(location);

% include "Temp" and "Dew Point"
all_temperature=zeros(1,num_location);

for index1=1:num_location,
    % For 1 digit temperatures, location-2 is the ">" sign
    % We extract only one position
    if temperature_web_text1(location(index1)-2)=='>'
        temperature=temperature_web_text1(location(index1)-1);
    end,

    % For 2 digit temperatures
    if (temperature_web_text1(location(index1)-2)~='>')&...
            (temperature_web_text1(location(index1)-3)~='1')
        temperature=temperature_web_text1(location...
            (index1)-2:location(index1)-1);
    end,

    % For 3 digit temperatures (more than 100 degree),
    % add one more digit
    if temperature_web_text1(location(index1)-3)=='1',
        temperature=temperature_web_text1(location(index1)...
            -3:location(index1)-1);
    end

    temp=str2num(temperature);
    all_temperature(index1)=temp;
end

% extract only next 48 hour Temperature data
num_Temperature=(num_location/2);
for index2=1:num_Temperature,
Temperature(index2)=all_temperature((2*index2)-1);
end

Temperature;

% locate the time frame
location_time=findstr(temperature_web_text1,':00 ');

% first location is irrelavant
location_time(1)=[];
num_location_time=length(location_time);

all_time_frame='';
for index2=1:num_location_time,

% in case, time has more than 1 digit
time_frame=temperature_web_text1(location_time(index2)-...
    2:location_time(index2)+5);
```

```
    if time_frame(1)=='>',
        time_frame(1)='0';
    end
    all_time_frame=strvcat(all_time_frame,time_frame);

end

all_time_frame;
% delete 0 infront of single digit hour
time_series_hour=datenum(all_time_frame);
% style 16
new_time_frame=datestr(time_series_hour,16);
```

REFERENCES

1. Donald F. Santa, J., *Welcome to the Hotel California: How Will the California Electricity Crisis Shape Federal Energy Policy?* The Electricity Journal 2001. **14**(6): p. 54-69.

2. McClintock, S.T. *California's Consumer-Power Crisis*. December 20, 2001 [cited; Available from: http://republican.sen.ca.gov/web/McClintock/article_print.asp?PID=206.

3. Douglas Caves, K.E.a.A.F., *Mitigating Price Spikes in Wholesale Markets through Market-Based Pricing in Retail Markets* The Electricity Journal 2000. **13**(3): p. 13-23.

4. Ilic, M., J. Black, and J. Watz, *Potential Benefits of Implementing Load Control.* IEEE Power Engineering Society Winter Meeting, 2002, 2002. **1**: p. 177-182.

5. *The California Electricity Crisis*. [cited; Available from: http://www.iscr.org.nz/f95,2419/2419_CA_electricity_crisis_JJ_2001.pdf.

6. *Chairwoman Dianne Jacob's Testimony Before the Federal Energy Regulatory Commission*. [cited; Available from: http://www.co.san-diego.ca.us/cnty/bos/sup2/press/speech-ferc-001114.html.

7. Yang, J., *A Market Monitoring System for the Open Electricity Markets.* Power Engineering Society Summer Meeting, 2001. IEEE, 2001. **1**: p. 235-240

8. Kirschen, D. and G. Strbac, *Fundamentals of Power System Economics*. 2004: John Wiley & Sons Inc., England.

9. Guan, X., Y.-C. Ho, and D.L. Pepyne, *Gaming and Price Spikes in Electric Power Markets.* IEEE Transactions on Power Systems, 2001. **6**(3): p. 402-408.

10. *Federal Energy Regulatory Commission, Assessment of Demand Response & Advanced Metering Staff Report*. August 2006.

11. *U.S. Department of Energy, Energy Policy Act of 2005*. [cited; Available from: http://www1.eere.energy.gov/femp/about/legislation_epact_05.html.

12. *U.S. Department of Energy, Benefit of Demand Response in Electric Markets and Recommendations for Achieving Them*. February 2006.

13. Kiesling, L. *The Role of Retail Pricing in Electricity Restructuring*. [cited; Available from: http://www.knowledgeproblem.com/kiesling_retail_chap.pdf.

14. Schuler, R., *Self-Regulating Markets for Electricity: Letting Customers into the Game.* IEEE PES Power Systems Conference and Exposition, 2004. **3**: p. 1524-1528.

15. *Electricity Markets Consumers Could Benefit from Demand Programs, but Challenges Remain*. [cited; Available from: http://www.gao.gov/new.items/d04844.pdf.

16. *Energy-Efficiency Funds and Demand Response Programs*. [cited; Available from: http://www1.eere.energy.gov/femp/program/utility/utilityman_energymanage.html.

17. *Georgia Power*. [cited; Available from: http://www.georgiapower.com/.

18. *NYISO*. [cited; Available from: http://www.nyiso.com/public/index.jsp.

19. *Gulf Power*. [cited; Available from: http://www.gulfpower.com/.

20. *CNT Energy*. [cited; Available from: http://www.cntenergy.org/.

21. *Ameren Illinois Utilities*. [cited; Available from: http://www.ameren.com/.

22. *Commonwealth Edison (ComEd)*. [cited; Available from: http://www.exeloncorp.com/ourcompanies/comed.

23. *Power Smart Pricing* [cited; Available from: http://www.powersmartpricing.org/index.php.

24. *Ameren's Real-Time and Estimated Electricity Prices*. [cited; Available from: https://www2.ameren.com/RetailEnergy/realtimeprices.aspx.

25. *Comverge*. [cited; Available from: http://www.comverge.com/.

26. *TheWattSpot*. [cited; Available from: http://www.thewattspot.com/.

27. Moholkar, A., *Computer Aided Home Energy Management System*, in *Department of Computer Science and Electrical Engineering*. 2005, West Virginia University.

28. Stoft, S., *Power System Economics, Designing Markets for Electricity*. 1 ed. 2002: Wiley-IEEE Press.

29. *The Electric Reliability Council of Texas*. [cited; Available from: http://www.ercot.com.

30. *Energy Information Administration: End-Use Consumption of Electricity*. [cited; Available from: http://www.eia.doe.gov/emeu/recs/recs2001/enduse2001/enduse2001.html#table2.

31. *Review of Residential Electrical Energy Use Data*. July 16, 2001, NAHB Research Center: Maryland.

32. Harper, S., D. Thurston, and P. Krein, *The Effects of Dynamic Residential Load Participation: Penetration Levels for Operational Impact on Reliability.* International Journal of Critical Infrastructures, 2007. **3**(1): p. 221-234.

33. P. Constantopoulos, F.S., and R. Larson, *ESTIA: A Real-time Consumer Control Scheme for Space Conditioning Usage under Spot Electricity Pricing.* Computers Operations Research, 1991. **19**(8): p. 751-765.

34. Black, J., *Integrating Demand into the U.S. Electric Power System: Technical, Economic, and Regulatory Frameworks for Responsive Load*, in *Engineering Systems Division*. 2005, Massachusetts Institute of Technology.

35. *Pacific Gas and Electric (PG&E) Company, Hourly Water Heating Calculations*. [cited; Available from: http://www.energy.ca.gov/title24/2005standards/archive/documents/2002-05-30_workshop/2002-05-17_WTR_HEAT_CALCS.PDF.

36. *MATLAB*. [cited; Available from: http://www.mathworks.com.

37. *Weather - Find Local Weather, National and International Weather Forecasts and Conditions*. [cited; Available from: www.findlocalweather.com.

38. *Thermal Environmental Conditions for Human Occupancy*. 1992, The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) Standard: ANSI/ASHRAE 55.

39. *Multi-Objective Optimization*. [cited; Available from: http://www-fp.mcs.anl.gov/otc/guide/OptWeb/multiobj.

40. *Priceline.com*. [cited; Available from: http://www.priceline.com.

41. *U.S. Department of Energy, Building America Performance Analysis Resources*. [cited; Available from: http://www.eere.energy.gov/buildings/building_america/pa_resources.html.

42. Chunduru, V., and Subramanian, N., *Effects of Power Lines on Performance of Home Control System.* International Conference on Power Electronics, Drives and Energy Systems (PEDES), 2006: p. 1-6.

43. Ishak, N., Din, N., Jamaludin, M., and Mohd, Y. , *Power Line Carrier Technology and Home Automation.* 2002 Student Conference on Research and Development Proceedings, Shah Alam, Malaysia, 2002: p. 505-508.

44. *X10 Home Automation*. [cited; Available from: http://www.x10.com.

45. *X10 Knowledgebase*. [cited; Available from: http://kbase.x10.com/wiki/Main_Page.

46. *CM15A 2-way USB Computer Interface*. [cited; Available from: http://www.activehomepro.com/.

47. *ActiveHome Professional: SDK, Software Development Kit*. [cited; Available from: http://www.activehomepro.com/sdk/sdk-info.html.

48. *Dimmable Lamp Module*. [cited; Available from: http://www.x10.com/automation/lm465_s.html.

49. *2-pin Polarized Appliance Module*. [cited; Available from: http://www.x10.com/automation/am486_s.html.

50. *RF Transceiver Module*. [cited; Available from: http://www.x10.com/automation/tm751_s.html.

51. *X10 Pro Appliance Module (3 Pin with AGC).* [cited; Available from: http://www.activehomepro.com/accessories/pro/pam02_wa1_s.html.

52. *TXB16 X10 Thermostat*. [cited; Available from: http://www.resconsys.com/products/stats/x10.htm.

53. *ERCOT Balancing Services Market Clearing Prices*. [cited; Available from: http://mospublic.ercot.com/ercot/jsp/balancing_services_mcp.jsp.

54. *Arlington TX Hourly Weather Forecast - Find Local Hourly Weather*. [cited; Available from: http://www.findlocalweather.com/by_the_hour/tx/arlington.html.

55. *Image Processing Toolbox - MATLAB*. [cited; Available from: http://www.mathworks.com/products/image/.

56. *Z-wave*. [cited; Available from: http://www.z-wave.com/modules/Z-Wave-Start/.

57. *Zigbee*. [cited; Available from: http://www.zigbee.org/en/index.asp.

58. *Zensys*. [cited; Available from: http://www.zen-sys.com/modules/Zensys/.

59. *TEMPer USB thermometer*. [cited; Available from: http://www.usbfever.com/index_eproduct_view.php?products_id=257.

60. *NI LabView*. [cited; Available from: http://www.ni.com/labview/.

61. *iPhone Developer Program*. [cited; Available from: http://developer.apple.com/iphone/program/.

BIOGRAPHICAL INFORMATION

Supun Tiptipakorn received the Bachelor Degree of Engineering in Electrical Engineering (BEEE), emphasis on electric power systems, from the King Mongkut's Institute of Technology Ladkrabang at Bangkok, Thailand; the Master of Business Administration (MBA), emphasis on management, from the University of Toledo, Ohio; and the Master of Science in Electrical Engineering (MSEE), emphasis on electric power systems, from the University of Wisconsin at Madison, Wisconsin. He pursued his PhD study from the Electrical Engineering Department of the University of Texas at Arlington, Texas and was a Graduate Research Assistant of the Energy Systems Research Center (ESRC). His research interests are the applications of system analysis and control techniques to power systems operation, protection, and planning. He is a member of Phi Beta Delta International Scholars Honor Society, Eta Kappa Nu National Electrical and Computer Engineering Honor Society, and Tau Beta Pi Engineering Honor Society.