Field Testing and Evaluation of a Wireless-Based Transit Signal Priority System

Final Report

Prepared by:

Chen-Fu Liao
Gary A. Davis

Minnesota Traffic Observatory
Department of Civil Engineering
University of Minnesota

CTS 11-25

Technical Report Documentation Page

1. Report No. CTS 11-25	2.	3. Recipients Accession No.
4. Title and Subtitle Field Testing and Evaluation of a Wireless-Based Transit Signal Priority System		5. Report Date October 2011 6.
7. Author(s) Chen-Fu Liao and Gary A. Davis		8. Performing Organization Report No.
9. Performing Organization Name and Address Department of Civil Engineering University of Minnesota 500 Pillsbury Drive, SE Minneapolis, MN 55455		10. Project/Task/Work Unit No. CTS Project #2009029 11. Contract (C) or Grant (G) No.
12. Sponsoring Organization Name and Address Intelligent Transportation Systems Institute Center for Transportation Studies University of Minnesota 200 Transportation and Safety Building 511 Washington Ave. SE Minneapolis, MN 55455		13. Type of Report and Period Covered Final Report 14. Sponsoring Agency Code
15. Supplementary Notes http://www.its.umn.edu/Publications/ResearchReports/		

16. Abstract (Limit: 250 words)

Most signal priority strategies implemented in various U.S. cities used sensors to detect buses at a fixed or preset distance away from an intersection. Traditional presence detection systems, ideally designed for emergency vehicles, usually send a signal priority request after a preprogrammed time offset as soon as transit vehicles are detected without the consideration of bus readiness. As part of the Urban Partnership Agreement, Metro Transit, Minnesota Department of Transportation (MnDOT), and the City of Minneapolis have implemented Transit Signal Priority (TSP) along Central Avenue from north Minneapolis (2nd Street SE) to south of I-694 (53rd Avenue NE) with total of 27 intersections. Transit performance before and after the deployment of a TSP strategy was examined through the data analysis process to evaluate the effectiveness and benefit of a TSP strategy. The objective of this study is to deploy and validate a wireless-based TSP strategy developed from earlier studies by considering bus schedule adherence, location and speed. A TSP onboard system using embedded computer was developed to interface with EMTRAC radio modules to bypass the EMTRAC TSP algorithm on current buses. Field experiments were performed by installing University of Minnesota (UMN) TSP units on four RTE10 buses for two weeks. The EMTRAC algorithm was temporary disabled on the test vehicles. Link travel time and node dwell time on the TSP-equipped route segments are compared. The results indicated the UMN TSP algorithm gain additional 3-6% of travel time reduction as compared to other RTE10 buses operating during the two-week test period.

17. Document Analysis/Descriptors		18. Availability Statement	
Public transit, Buses, Traffic signal preemption, Wireless communication systems, Route model, Routes		No restrictions. Document available from: National Technical Information Services, Alexandria, Virginia 22312	
19. Security Class (this report) Unclassified	20. Security Class (this page) Unclassified	21. No. of Pages 96	22. Price

Field Testing and Evaluation of a Wireless-Based Transit Signal Priority System

Final Report

Prepared by:

Chen-Fu Liao
Gary A. Davis

Minnesota Traffic Observatory
Department of Civil Engineering
University of Minnesota

October 2011

Published by:

Intelligent Transportation Systems Institute
Center for Transportation Studies
200 Transportation and Safety Building
511 Washington Avenue S.E.
Minneapolis, Minnesota 55455

ACKNOWLEDGMENTS

We would like to thank the Intelligent Transportation Systems (ITS) Institute and Center for Transportation Studies (CTS) at University of Minnesota for supporting this project. The ITS Institute is a federally funded program administrated through the Research and Innovative Technology Administration (RITA), U.S. Department of Transportation (USDOT). We also would like to recognize the following people and organizations for their invaluable assistance in making this research possible.

- Varun Ramesh & Rahul Iyer, electrical engineering graduate students, for their support on embedded programming and field testing.
- Metro Transit, for providing bus data and invaluable support on field testing and data collection.
- EMTRAC, for providing interface protocol to access the GPS and wireless radio.
- The Minneapolis Department of Public Works, for allowing us to perform testing with its signal controller and roadside unit through wireless communications.
- Minnesota Department of Transportation (MnDOT), for providing traffic signal timing and phasing data
- Minnesota Traffic Observatory, Department of Civil Engineering, for using the lab facility.

TABLE OF CONTENTS

LIST OF ACRONYMS AND ABBREVIATIONS

ABSP	Adaptive Bus Signal Priority
ACC	Adaptive Cruise Control
ADCS	Automatic Data Collection System
AFC	Automatic Fare Collection
APC	Automatic Passenger Counter
AVL	Automatic Vehicle Location
CAPRI	Categorized Arrival-based Phase Reoptimization at Intersection
CBD	Central Business District
CCA	Cooperative Collision Avoidance
CF	Compact Flash
CICAS	Cooperative Intersections Collision Avoidance Systems
COOPERS	Co-operative Systems for Intelligent Road Safety
COTS	Commercial-Off-The-Self
CTA	Chicago Transit Authority
CTS	Center for Transportation Studies
CVIS	Collaborative Vehicle Infrastructure System
DSRC	Dedicated Short Range Communications
DT	Dwell Time
EV	Emergency Vehicle
EVP	Emergency Vehicle Preemption
FHWA	Federal Highway Administration
FS	Farside Bus Stop
ft	Feet
GIS	Geographic Information System
GPS	Global Positioning System
GUI	Graphical Users Interface
HCM	Highway Capacity Manual
IQR	Inter-Quartile Range
ITS	Intelligent Transportation Systems
ITSA	Intelligent Transportation Systems America
LACMTA	Los Angeles County Metropolitan Transportation Authority
LIDAR	Light Detection and Ranging
LOS	Level Of Service
LT	Left Turn
min	Minute
MnDOT	Minnesota Depart of Transportation
MT	Metro Transit
MTA	Metropolitan Transit Authority
MUTCD	Manual on Uniform Traffic Control Devices
NAND	N-AND type flash memory
NB	Northbound
NS	Nearside Bus Stop
NYCT	New York City Transit
OBU	Onboard Unit
OD	Origin to Destination

OTP	On-Time Performance
PATH	Partners for Advanced Transit and Highways
RADAR	Radio Detection and Ranging
RF	Radio Frequency
RHODES	Real-time Hierarchical Optimized Distributed Effective System
RITA	Research & Innovative Technology Administration
RSU	Roadside Unit
RT	Right Turn
RTE	Route
SATA	Serial Advanced Technology Attachment
SB	Southbound
SBC	Single Board Computer
SCOOT	Split Cycle Offset Optimization Technique
SD	Secure Digital
sec	Second
SMART-Signals	Systematic Monitoring of Arterial Road Traffic and Signals
SPLIT	Selective Priority to Late buses Implemented at Traffic signals
SQL	Structured Query Language
STDEV	Standard Deviation
TCC	Transit Control Center
TCRP	Transit Cooperative Research Program
TH	Truck Highway
TMC	Traffic Management Center
TOD	Time of Day
TP	Time Point
TT	Travel Time
TSP	Transit Signal Priority
UMN	University of Minnesota
UPA	Urban Partnership Agreement
USDOT	U.S. Department of Transportation
VANET	Vehicular Ad-Hoc Network
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
VII	Vehicle Infrastructure Integration
VCU	Vehicle Control Unit
WLAN	Wireless Local Area Network

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Most signal priority strategies implemented in various U.S. cities used sensors to detect buses at a fixed or preset distance away from an intersection. Traditional presence detection systems, ideally designed for emergency vehicles, usually send a signal priority request after a preprogrammed time offset as soon as transit vehicles are detected without the consideration of bus readiness. Significant amount of research projects focusing on various intelligent transportation systems (ITS) applications under the connected-vehicle research, previously called *IntelliDrive*™ or Vehicle Infrastructure Integration (VII) framework, have been investigated since the introduction of the initiative. Data collected through the Dedicated Short Range Communications (DSRC) network could potentially enable hundreds of possibilities including safety, mobility, and commercial uses, from intersection collision avoidance and dynamic route guidance to road-level weather advisories and electronic toll collection.

Wireless communications systems have made rapid progress in the past decade and are commercially available. For example, Los Angeles County Metropolitan Transportation Authority (MTA) implemented wireless technology in a transit signal priority system along a corridor using the IEEE 802.11b protocol. The wireless system on each bus sends an Internet protocol (IP) addressable message to an access point that covers three to four intersections. A wireless client installed in the signal cabinet communicates with a modified traffic controller to request signal priority. King County MTA in Washington is also planning to design a wireless transit signal priority (TSP) system similar to the implementation in Los Angeles County. Pace, a suburban bus division of the Regional Transportation Authority, provides bus services throughout Chicago's six-county suburban region. Pace recently was awarded with several research projects to deploy bus rapid transit (Cermak, Golf Road, and south suburban) and transit signal priority (Cicero Avenue and Rand Road) through the Safe, Accountable, Flexible, Efficient Transportation Equity Act: A Legacy for Users (SAFETEA-LU) bill. Pace is also working with a consulting company to identify wireless-based systems to provide bus signal priority and to report priority status and system performance back to its transit operation center.

As part of the Urban Partnership Agreement (2008), Metropolitan Council (Met Council), Minnesota Department of Transportation (MnDOT), and the City of Minneapolis have implemented TSP along Central Avenue. Bus route 10 and 59 served along the corridor in parallel of Interstate 35W from north Minneapolis (2nd Street SE) to south of I-694 (53rd Avenue NE) with total of 27 intersections (about 5.5 miles). The TSP solution is provided by EMTRAC Systems based in Illinois (http://www.emtracsystems.com/). Transit performance before and after the deployment of a TSP strategy was examined through the data analysis process to evaluate the effectiveness and benefit of a TSP strategy. Bus schedule can potentially be adjusted to take advantage of a TSP strategy through running time and Time Point (TP) time analysis.

The objective of this study is to deploy and validate a wireless-based TSP strategy developed from earlier study that considers bus schedule adherence, location and speed for priority request. A TSP onboard system, namely UMN TSP system, using embedded Single Board Computer (SBC) was developed to interface with EMTRAC radio modules to bypass the EMTRAC TSP algorithm on current buses. The goal is to take advantage of the already instrumented onboard equipment and roadside infrastructure for buses to communicate with intersection signal

controllers when they are ready to request signal priority. Performance evaluations of our TSP algorithm as compared to existing TSP operation were analyzed using a transit data processing framework developed by Liao & Liu (2010). The methodological data-processing framework can process a massive amount of transit data, including vehicle location, passenger count, and electronic fare transactions collected from existing automatic data collection system (ADCS) such as automatic vehicle location (AVL), automatic passenger count (APC) and automatic fare collection (AFC).

Field experiments were performed by installing University of Minnesota (UMN) TSP units on four RTE10 buses for two weeks. The EMTRAC algorithm was temporary disabled on the test vehicles. Link travel time and time point (TP) time on the TSP-equipped route segments are compared. The results indicated the UMN TSP algorithm gain additional 3-6% of travel time reduction as compared to other RTE10 buses operating during the two-week test period. This report documents the development, verification and validation of the TSP algorithm and field testing results. The developed data analysis framework (Liao & Liu; 2010) can assist a number of research and applications such as transit route performance measurement, TSP and decision-making support for transit planning and operation. The vision is to use the data processing framework to generate deployment suggestions for TSP deployment. The future plan is to continue ongoing collaboration with Metro Transit to develop tools to support transit operation, scheduling and planning.

1 INTRODUCTION

Minnesota is one of five communities nationwide to receive funding through the U.S. Department of Transportation's Urban Partnership Agreement (UPA) program to develop strategies and to implement and deploy applications that will reduce traffic congestions in the Twin Cities. As part of the UPA program, Twin Cities Metropolitan Council (Met Council) and the City of Minneapolis have instrumented Metro Transit buses and signalized intersectioncontrollers with transit signal priority (TSP) capability along Central Avenue (Route 10 – local service and Route 829/59 – limited stop express service) running in parallel with Interstate 35W. The EMTRAC Systems (http://www.emtracsystems.com/) was selected to implement wireless TSP on over 700 Metro Transit (MT) buses and 27 signalized intersections along the Central Avenue in north of Minneapolis. The wireless-based TSP approach allows late buses to request for green signal while approaching a signalized intersection.

Signal priority for transit has been studied and proposed as an efficient way to improve transit travel & operation. Bus signal priority has been implemented in several U.S. cities to improve schedule adherence, reduce transit operation costs, and improve customer ride quality. Signal priority strategies have helped reduce the transit travel time delay, as discussed in the literature (ITSA, 2002), but the transit travel time reduction varies considerably across studies (Collura et. al, 2003). Signal priority and preemption are often used synonymously; however, they are different processes. Signal preemption is traditionally used for Emergency Vehicles (EV) or at railroad crossing. Preemption interrupts normal intersection signal process to provide high priority for special events. Signal priority modifies the normal signal operation in order to accommodate better service for transit vehicles (ITSA, 2004).

1.1 Background

Current signal priority strategies implemented in various U.S. cities mostly utilized sensors to detect buses at a fixed or at a preset distance away from the intersection. Signal priority is usually granted at a preprogrammed time offset, after detection. Engineers usually have to adjust the detector location, receiver line of sight and timing offset for each intersection in order to ensure its effectiveness. These TSP strategies do not necessary consider the bus's speed and its distance from intersection when determining the appropriate time to request signal priority. Previously proposed study incorporated the GPS/AVL system on the buses in Minneapolis and developed a signal priority strategy based on the bus's timeliness with respect to its schedule, number of passengers, location and speed of a bus.

Wireless communications systems have made rapid progress and are commercially available. Bus information (e.g. speed, location, number of passengers, bus ID) can be transmitted wirelessly to a traffic controller or to a regional Traffic Management Center (TMC) in making decisions for signal priority. There are several wireless communication systems installed on each bus under the current Metro Transit setup. An 800-MHz Motorola digital voice radio is used for communication between bus driver and Transit Control Center (TCC). Another 800-MHz analog data radio is used to poll bus location and passenger count data every minute. A Wireless Local Area Network (WLAN) 802.11x is also installed on the buses. This is used to upload/download files between the bus and the TCC central server, when the bus is within the proximity of the transit garage.

Our previous study developed a wireless-based transit signal priority prototype and evaluated Wi-Fi and Dedicated Short Range Communication (DSRC) wireless technologies. Using the Wi-Fi network for TSP application can certainly reduce cost by taking advantage of the existing infrastructure. However, availability of data bandwidth and quality of service, concern of network reliability and data security need to be addressed when choosing the Wi-Fi technology. The DSRC radio is potentially good with excellent performance (short range with fast data communication rate), but the availability of DSRC is currently limited. We certainly don't know whether there will be national "rollout".

1.2 Research Objectives

We hope to provide effective signal priority to buses with minimal impact on other traffic using the already equipped GPS/AVL system on the bus. The GPS system offers better information regarding bus trajectory than the presence detection sensors generally used while requesting for traffic signal priority. The objective of this study is to deploy and validate a wireless-based TSP strategy developed from earlier studies (Liao & Davis, 2008) by considering the bus schedule adherence, location and speed. A TSP onboard system, namely UMN TSP, using embedded Single Board Computer (SBC) was developed to interface EMTRAC radio module to bypass the EMTRAC TSP algorithm. The goal is to take advantage of the already instrumented onboard and roadside infrastructures for buses to communicate with intersection signal controllers when it's ready to request for priority.

1.3 Literature Review

The European Collaborative Vehicle Infrastructure System (CVIS) aims to develop a cooperative intelligent system to enable Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communications. In the U.S., the connected vehicle research, previously called IntelliDrive, is to deploy a nationwide communications network along the national roadways that enables communications between vehicles and roadside infrastructure to improve transportation and quality of life. The report from Federal Highway Administration (FHWA) documents the IntelliDrive architecture and its design requirements (Farradyne, 2005). Significant amount of research projects focusing on various intelligent transportation systems (ITS) applications under the IntelliDrive framework have been investigated since the introduction of the IntelliDrive initiative. Data collected through the IntelliDrive network could potentially enable hundreds of possibilities including safety, mobility, and commercial uses, from intersection collision avoidance and dynamic route guidance to road-level weather advisories and electronic toll collection. For example, Wischhof et al. (2003) and Zhang (2003) investigated the dissemination of vehicle-based traffic and travel information through the Vehicle Infrastructure Integration (VII) network. Wu et al. (2005) evaluated the efficiency of message propagation between vehicles through simulation. Xu et al. (2002) study the effect of vehicle vehicle/vehicle-roadside communication on the performance of adaptive cruise control (ACC) systems through simulations. Collision avoidance technologies that use the VII infrastructure are also being developed as part of the Cooperative Intersections Collision Avoidance Systems (CICAS, http://www.its.dot.gov/cicas/index.htm) initiative. Biswas et al. (2006) presented the concept of Cooperative Collision Avoidance (CCA) and its implementation requirements in the context of the vehicle-to-vehicle wireless network. Alexander et al. (2006) designed and deployed a transportable rural intersection surveillance system encompassing RADAR (Radio Detection and

Ranging), LIDAR (Light Detection and Ranging), camera systems and wireless communications between infrastructure and vehicles to investigate the gap acceptance behavior of drivers at rural intersections.

Wireless communications systems have made rapid progress in the past decade and are commercially available. McNally et al. (2003) developed an in-vehicle, GPS-based system to provide real-time vehicle guidance information through wireless communication technologies. Fitzmaurice (2005) reviewed the recent technology advances and regulatory changes that have encouraged the mobile wireless applications in rail and urban transit environments. Torrent-Moreno et al. (2004) investigated a study on the probability of reception of a broadcast message by another car and how to provide priority access for important warnings in 802.11-based vehicular ad-hoc networks (VANET). One of the key usages of VANET is to support vehicle safety applications through the broadcast operations for informing the immediate neighboring vehicles. Stibor et al. (2007) evaluated the number of communication partners in communication range and maximum communication duration for a vehicular ad-hoc network using the IEEE 802.11p transceivers in a highway scenario. Marca (2006) performed testing and benchmarked possible throughput of 802.11b wireless communication technology for vehicle to roadside infrastructure communications. Böhm et al. (2008) evaluated different wireless communication technologies, including broadcast, cell based and dedicated short range technologies, and their effectiveness of transmitting road-safety relevant information from infrastructure to vehicle (I2V) as part of the Co-operative Systems for Intelligent Road Safety (COOPERS) program co-funded by the European Commission. Ahmed et al. (2008) developed a blue tooth and wireless mesh networks platform for traffic network monitoring. The platform uses traveling cars as data collecting sensors or probes and uses wireless municipal mesh networks to transmit collected traffic data.

Los Angeles County Metropolitan Transportation Authority (LACMTA) has implemented wireless technology in a transit signal priority system along several corridors using the IEEE 802.11b protocol (Kittleson & Associates, 2006). The wireless system on each bus sends an Internat protocol (IP) addressable message to an access point that covers three to four intersections. A wireless client installed in the signal cabinet communicates with a modified traffic controller firmware to request signal priority. TSP request was sent wirelessly from bus onboard unit to an access point connected to traffic controller through a serial (RS-232) interface. Bus messages and TSP responses from controller were also collected and sent to central control via cellular communication. Iteris, Inc. (www.iteris.com) was recently selected by LACMTA for the design, acquisition, deployment, and ongoing operation and maintenance of bus traffic signal priority systems at 211 signalized intersections maintained by18 local agencies along Manchester Boulevard, Garvey Boulevard/Cesar Chavez Avenue, and Atlantic Boulevard. King County MTA in Washington is also planning to design wireless TSP system similar to the implementation in LA County.

Pace, the suburban bus division of the Regional Transportation Authority, provides bus services throughout Chicago's six-county suburban region. Pace recently awarded several research projects to deploy bus rapid transit (Cermak, Golf Road, and south suburban) and transit signal priority (Cicero Avenue and Rand Road) through the SAFETEA-LU bill (http://www.pacebus.com/sub/vision2020/ federal_projects.asp). Pace is working with a consulting company to identify a wireless-based system to provide signal priority to buses and report status and performance back to the transit operation center.

Signal priority requests for transit or emergency vehicles can potentially be sent to the signal controller through the vehicle-to-infrastructure communication architecture as described in VII architecture (Farradyne, 2005). Signal priority for transit vehicles has been studied and proposed as an efficient way to improve transit travel time and schedule adherence, to reduce transit operation costs, and to improve customer riding quality. Signal priority strategies have helped reduce the transit travel time delay, as discussed in the literature (ITS America, 2002), but the transit travel time reduction varies considerably across studies (Collura et al., 2006). Unlike signal preemption, which interrupts the normal intersection signal process to provide high priority for special events (emergency vehicle or railroad crossing), TSP modifies the normal signal operation in order to accommodate better service for transit vehicles (ITS America, 2004).

Transit signal priority has been implemented in several U.S. cities (Seattle, Portland, Los Angeles, and Chicago) as well as in Europe. Various technologies have been deployed for bus priority including Opticom™ (St. Cloud, 2000), inductance loop detectors (Los Angeles), and RF tag (Seattle, King County, 2002). Recently, Crout (2005) at Tri-County Metropolitan Transportation District of Oregon (TriMet) proposed two types of analyses (corridor and intersection level) to evaluate the effectiveness of the TSP effort on transit operations over 300 signals implemented with signal priority. Current signal priority strategies implemented in various U.S. cities mostly utilize sensors to detect buses at a fixed or at a preset distance away from the intersection. Signal priority is usually granted after a preprogrammed time-offset, after detection. Engineers usually have to adjust the detector location, receiver line of sight and timing offset for each intersection in order to ensure its effectiveness. Liu et al. (2004) presented a theoretical model to quantitatively address the relation between bus detector location and effectiveness of transit signal priority systems. Li et al. (2007) proposed an active signal priority optimization model for Light-Rail Transit (LRT) in a simulation study by estimating the train travel and dwell time. Most TSP strategies do not consider the transit's speed and its distance from the intersection when determining the appropriate time to request signal priority.

Metro Transit in the Twin Cities Metro Area (http://www.metrotransit.org/) previously performed an evaluation to provide signal priority for buses on Lake Street in Minneapolis using 3M Opticom™ systems. A special software modification was made to provide transit priority using green extension and red truncation strategies. However, the Opticom™ system, ideally designed for emergency vehicle preemption (EVP), was not able to adjust the trigger timing for buses approaching nearside bus stops, and buses often missed the priority green period when they were ready to depart. Since several intersections along Lake Street were already operating at their capacity, the potential for providing transit priority without delaying vehicle traffic was somewhat constrained. There were also issues of buses traveling across different municipalities that were unwilling to provide signal priority for transit. Results from this previous evaluation study were not promising. With the installation of GPS system on its fleet, Metro Transit now constantly monitoring bus locations in relation to their schedules, in order to provide more reliable transit services and enhance transit operation and management. Bus location, travel time, delay and other traffic information can also be collected and integrated to assist traffic operation management and to inform the traveling public. Metro Transit would like to use the already installed GPS/AVL system as the basis of a transit-based intelligent transportation system (ITS).

Bus information (e.g. speed, location, number of passengers, bus ID) can be transmitted wirelessly to a traffic controller or to a regional Traffic Management Center (TMC) in making decisions to grant signal priority request. Several wireless communication systems were installed on Metro Transit buses as standard system configuration. An 800-MHz Motorola digital voice radio is used for communication between bus driver and Transit Control Center (TCC). Another 800-MHz analog data radio is used to poll bus location and passenger count data every minute. A Wireless Local Area Network (WLAN) 802.11x is also installed on the bus to automatically upload or download data files between the bus computer and the central server at TCC when the bus is within the proximity of the transit garage.

Researchers at California PATH (Partners for Advanced Transit and Highways) studied an "Adaptive Bus Signal Priority" (ABSP) to apply an active priority strategy for buses, by including bus GPS information, traffic detector data, and a travel-time predictor to an adaptive model (Liu et al., 2003). Wadjas and Furth (2003) developed a methodology by adapting advanced detection and cycle length to provide transit signal priority. The adaptive control includes traffic density and queue length estimation in a simulation study. Skabardonis and Geroliminis (2008) developed an analytical model for real-time estimation of arterial travel time. A signal priority algorithm, extends the active signal priority strategy initially proposed by Skabardonis (2000), was developed and incorporated into their base model to provide system wide adjustments to the signal timing plans and priority based on the real-time traffic information. Li et al (2005) proposed a heuristic TSP algorithm to provide signal priority to buses as well as limit negative impact on cross-street traffic. Traditional TSP strategies implemented in United States are mostly fixed-location detection systems and implemented with time-of-day signal control systems. TSP systems using fixed-location detection usually do not work well with nearside bus stops, due to the uncertainty in bus dwell time. Zheng et al. (2007) developed a theoretical model to estimate the corresponding delays at nearside bus stops. Kim and Rilett (2005) proposed a weighted least squares regression model in simulation to estimate bus dwell time in order to overcome nearside bus stop challenges. Ghanim et al. (2007) developed an artificial neural network modeling tool to predict intersection bus arrival time on approaches with nearside bus stops. Rakha et al. (2006) performed field and simulation evaluation along U.S. Route 1 corridor. They recommended further consideration on existence of queues in transit signal priority strategy and suggested no nearside bus stop implementation. Furth and SanClemente (2006) investigated the impact of bus stop location on bus delay. They found far-side bus stops cause small reduction in delay or no effect. Nearside bus stops more often increase bus delay.

A bus priority algorithm could also be integrated into an adaptive intersection signal control model. Research based on the bus priority facilities available within the Split Cycle Offset Optimization Technique (SCOOT) traffic signal control system was conducted by Bretherton et al. (1996). Traffic signal priorities can be controlled by a central SCOOT computer or by a local traffic signal controller. A local controller can achieve faster TSP response to buses than a centralized control. Different strategy options for providing bus priority at signals are compared by McLeod & Hounsell (2003) using the simulation model called Selective Priority to Late buses Implemented at Traffic signals (SPLIT). McLeod suggested that differential (conditional) priority strategies (e.g. granting priority for lateness) give the best results, as these provide a good balance between travel time and passenger waiting time. Furth and Mueller (2000) conducted a field study with three priority conditions (no priority, absolute priority, and conditional priority) at a transit route in the Netherlands. The study found absolute priority caused large delays to other traffic while conditional priority caused little, if any additional delay. Dion and Rakha (2005) developed a simulation approach to integrate TSP within an adaptive traffic control system. They evaluate three different signal control scenarios and found adaptive signal control reduced negative impacts on general traffic while providing signal priority to buses. Recently, Mirchandani & Lucas (2004) developed a Categorized Arrival-based Phase Reoptimization at Intersection (CAPRI) strategy that integrates transit signal priority within a real-time traffic adaptive signal control system, called RHODES (Real-time Hierarchical Optimized Distributed Effective System, 2001). “Weighted bus” and “phase constrained” approaches were developed for providing transit priority through the RHODES-CAPRI framework. Mirchandani et al. (2001) proposed a hierarchical optimization approach where traffic signals are determined by considering delays of all vehicles on the network as well as bus passenger counts and schedule while providing transit priority (RHODES/BUSBAND).

As transit agencies are more actively seeking solutions to traffic congestion, such as signal priority and various traffic management schemes, they need tools to monitor whether countermeasures are effective. For example, a Portland State University study conducted for Tri-Met using archived AVL/APC data found that while signal priority reduced running time on some routes, it had no positive effect on others (Kimpel et al., 2005). Altun and Furth (2009) proposed a methodology to adjust bus schedules to take advantage of transit signal priority in a simulation study. Bus schedules can potentially be adjusted to take advantage of a TSP strategy through proposed data analysis. In addition, arterial traffic information, intersection signal timing, and weather condition (Hofmann and O’Mahony, 2005) can also be included in the transit processing framework to gain a holistic view of transit performance network-wide.

Transit performance analysis can also include other data that may significantly affect the bus travel time, such as latest traffic information and/or historical traffic patterns. The authors recently developed a SMART-SIGNAL (Systematic Monitoring of Arterial Road Traffic and Signals) system to evaluate arterial travel time on 11 consecutive intersections (Liu et al., 2009). The transit database model can later be integrated with the arterial travel time model to develop robust and effective operation plans and to support other intelligent transit applications. For example, Cathey and Dailey (2003) developed a model using Kalman filter and AVL data to predict transit vehicle arrival and departure time. Travel time estimation on arterial streets has been a challenging task for traffic engineers because of the interrupted nature of urban traffic flows. Many research efforts have been devoted on this topic, but their successes are limited due to the availability of traffic data from signalized intersections. Berkow et al. (2008) developed a model to estimate arterial travel time by combining the loop detector data and bus AVL data. Additional features to include arterial traffic data (for example, signal timing/phasing, arterial traffic volume, speed, delay, and travel time) can be incorporated for transit applications as shown in Figure 6-1. These applications include real-time bus arrival time prediction, running time estimation, schedule adjustments, recommendations for TSP deployment, and real-time service management in the future. Data visualization based on system performance can also be used as an effective tool to support transit planning, scheduling and decision making.

1.4 RTE10 Signal and Service Timeline

Timeline of traffic signal changes and bus service changes due to TSP implementation on Central Avenue is listed in Table 1.1. Traffic signals between 2nd and 27th Avenue were changed from pre-timed to semi-actuated in July 2009. Bus running time was reduced by about 2 minutes on all trips and service types along the corridor. RTE 10 weekday schedule was adjusted to coordinate with NorthStar commuter rail schedule in December 2009. Minimum of 10-minute headway was initially applied for TSP activation in Minneapolis between 2nd St. and 37th Ave. Bus lateness threshold was changed from 10-min to 5-min in June 2010. TSP request settings were changed to 3-min lateness and 8-min headway threshold in November 2010. Detailed intersection signal timing information is included in Appendix A.

Table 1.1. Timeline of Signal Changes and Transit Services

Description of Changes Related to Route 10	Signals	Service
Signals changed from pre-timed to semi-actuated in Minneapolis. (between 2nd and 27th)	7/1/2009	
Traffic signal timings implemented in Minneapolis (for general traffic)	10/1/2009	
Traffic signal timings implemented by MnDOT and Ramsey County (for general traffic)	12/1/2009	
TSP Signal Timing Parameters installed on all signals	12/1/2009	
Running time reduction of about 2 minutes on all trips and service types along corridor. Weekday peak route 10 adjusted to coordinate with NorthStar commuter rail schedule.		12/12/2009
Initial date when TSP was active for Route 10 (and at Roseville intersections)	1/1/2010	
10 min. headway limitation for TSP activation in Minneapolis. (2nd St. to 37th Ave.)	2/25/2010	
Nicollet Mall changed to two-block spacing. Rerouted past Convention Center. Route 829 renumbered to 59. Beginning of 10/59 coordination - 8 trips added to 59		3/20/2010
Peak 10 and 59 schedules coordinated - 6 Rte 10 trips converted to Rte 59		5/15/2010
Change to 5 min. late threshold for all signals	6/1/2010	
Running time increased downtown and beyond 53rd. 3 School trips added to the weekday 10		9/11/2010
Change to 3 min late & 8-min headway threshold for all signals	11/26/2010	

1.5 Report Organization

This report is organized as follows. Transit signal priority strategy is presented in Chapter 2. Development of signal priority system and system components are discussed in Chapter 3. Field experiments and data collections are discussed in Chapter 4. In Chapter 5, experiment results and data analyses are compared and presented. Finally, Chapter 6 discussed and summarized the findings.

Signal timing and phasing are included in Appendix A. Traffic volume of RTE10 links are included in Appendix B. More technical document of the embedded SBC is included in Appendix C. Appendix D includes list of RTE10 stops and signalized intersections. Data processing Structured Query Language (SQL) scripts are listed in Appendix E. And, sample source code is included in Appendix F.

2 ADAPTIVE BUS SIGNAL PRIORITY STRATEGY

To illustrate our priority strategy, consider a simple eastbound/westbound corridor as shown in Figure 2.1. For a bus approached a bus stop or signalized intersection, there are basically two scenarios, a nearside bus stop or a far-side bus stop. For the nearside bus stop, a bus will stop for boarding/alighting before passing the signalized intersection, as illustrated in Figure 2.1 by the eastbound bus approaching stop j and intersection *i*. Estimated bus dwell time at the nearside bus stop needs to be considered by the signal controller to provide signal priority to the bus in a timely manner. For the far-side bus stop, a bus passes through the intersection first before its arrival at the stop (see Figure 2.1 westbound bus approaching intersection *i* and bus stop *k*). Bus travel time to the intersection needs to be considered when providing priority.

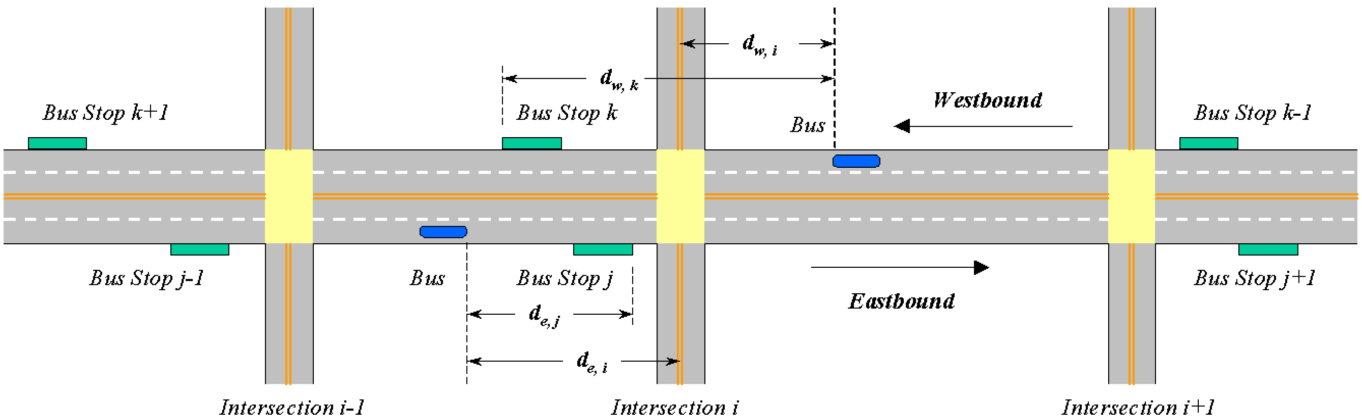

Figure 2.1. Example of an East-West Corridor for Signal Priority

2.1 Bus Stop Location Consideration

According to the findings from literature, Farside (FS) stop benefits more from TSP treatment that the Nearside (NS) does. Suggestions were recommended to move NS stop to FS for better TSP performance. However, relocating bus stop usually generates resistances from local community and often becomes a political issue. Most of bus stops are NS in Minnesota. Snow banks piled up along the street over the long winter months. Shoveling snow at FS or mid-block bus stops create extra maintenance effort to ensure accessibility to all users.

2.1.1 Nearside (NS) Bus Stop

Consider the bus traveling in the eastbound as shown in Figure 2.1. Expected bus dwell time, T_{dj}, at bus stop j can be forecasted using historical dwell time statistics. Expected bus travel time, T_{aj}, from its current location to bus stop can be calculated via,

$$T_{aj} = \frac{d_{e,j}}{v_b} + T_{br} + T_{delay} \tag{2- 1}$$

Where,

v_b: is bus speed,

$d_{e,j}$: is the distance from the current bus location to bus stop *j*,

T_{br} : is bus braking/stopping time, and

T_{delay} : is the traffic delay on bus route.

The expected bus travel time (T_{ji}) from bus stop j to intersection *i* can also be calculated as follows, assuming the distance from the nearside bus stop to the intersection is relatively short compared to the distance needed to accelerate to running speed.

$$T_{ji} = \sqrt{\frac{2(d_{e,i} - d_{e,j})}{a}} + T_{bc} \tag{2-2}$$

Where,

$d_{e,i}$: is the distance from eastbound bus to intersection *i*,

$d_{e,j}$: is the distance from eastbound bus to bus stop j,

a : is the bus acceleration, and

T_{bc} : is the bus clearance time.

Therefore the predicted time at which the eastbound bus passes intersection *i* can be calculated as follows.

$$\hat{t}_{ei} = t + T_{aj} + T_{dj} + T_{ji} \tag{2-3}$$

Where,

t : is the current time, sec.

And estimated time for the bus leaving stop *j* is,

$$\hat{t}_{lj} = t + T_{aj} + T_{dj} \tag{2-4}$$

The desired signal priority request should then be sent at δ_n seconds prior to the bus departure time at stop *j*. That is, at time $\hat{t}_{lj} - \delta_n$, where

$$\delta_n = t_{cp} + t_{comm} + t_{const} \tag{2-5}$$

t_{cp} : is the controller processing time,

t_{comm} : is the communication latency time, and

t_{const} : is an additional time constant.

The signal priority service should be ended at $\hat{t}_{ei} + T_{xi}$, where T_{xi} is the time for the bus to cross intersection *i*. If both beginning ($\hat{t}_{lj} - \delta_n$) and ending ($\hat{t}_{ei} + T_{xi}$) of the estimated priority service fall within the green split, no action needs to be taken at the controller. If $\hat{t}_{lj} - \delta_n$ falls in the green split and $\hat{t}_{ei} + T_{xi}$ falls in the red split, extended green time is needed to ensure that bus could pass the intersection. However, if the estimated beginning of priority service time ($\hat{t}_{lj} - \delta_n$) falls within the red light period, red signal truncation or early green light treatment is needed to provide bus signal priority.

2.1.2 Far-side (FS) Bus Stop

For a bus approaching an intersection prior to its arrival at next bus stop, for example, the bus traveling in westbound as shown in Figure 2.1, signal priority should be provided based on bus traveling speed and traffic conditions. The estimated time (T_{ai}) to arrive at intersection i can be calculated as,

$$T_{ai} = \frac{d_{w,i}}{v_b} + T_{delay} \tag{2-6}$$

Where,

$d_{w,i}$: is the distance from westbound bus to intersection *i*,

v_b : is bus speed, and

T_{delay} : is the traffic delay on bus route.

Therefore the estimated time for westbound bus passing intersection *i* can be calculated as follows.

$$\hat{t}_{wi} = t + T_{ai} \tag{2-7}$$

Where,

t : is the current time, sec.

The desired signal priority would need to begin at δ_n seconds prior to the bus arriving intersection *i* ($\hat{t}_{wi} - \delta_n$), where δ_n is defined as equation (5). The signal priority service can be ended at $\hat{t}_{wi} + T_{xi}$, where T_{xi} is the time for bus to cross intersection *i*. If both beginning ($\hat{t}_{wi} - \delta_n$) and ending ($\hat{t}_{wi} + T_{xi}$) of the estimated priority service fall within the green split, no action needs to be taken by the controller. If $\hat{t}_{wi} - \delta_n$ falls in the green split and $\hat{t}_{wi} + T_{xi}$ falls in the red split, extended green time is need to ensure bus could pass the intersection. However, if the estimated beginning of priority service time ($\hat{t}_{wi} - \delta_n$) falls within the red light period, red signal truncation or early green light treatment is needed to offer bus priority.

2.2 Estimation of Bus Dwell Time at Bus Stop

Ghanim et al. (2007) developed an artificial neural network modeling tool to predict the bus arrival time on approaches with nearside bus stops based on observed travel time, boarding and alighting demand, and current traffic condition. We used a simpler linear regression model to predict dwell time based on the number of boarding and alighting passengers, average headway between buses, schedule adherence, number of door on the vehicle, fare collection method, and bus type. Estimated passenger arrival rates will be used to forecast bus dwell time at each stop. Based on the collected data, we assume the passenger arrivals at each stop follow a Poisson distribution with an arrival rate, λ, calculated from the mean of the collected passenger arrival rate. A Poisson process subroutine was developed to generate numbers of passengers boarding and alighting at each stop during the simulation.

Bus dwell time at a bus stop for boarding can be estimated using the following equation.

$$T_{dj}^{B} = \lambda_j(t) \times \left[t_k(j) - t_{k-1}(j)\right] \times T_{boarding} + n_a \times T_{alighting} + T_{ab} \tag{2-8}$$

Where,

T_{dj}^{B} is the bus dwell time for boarding at stop j,

$\lambda_j(t)$ is the passenger arrival rate for stop j,

$t_k(j)$ is the arrival time of bus k at stop j,

$t_{k-1}(j)$ is the arrival time of bus k-1 at stop j,

$T_{boarding}$ is the average boarding time per passenger,

n_a is the number of alighting,

$T_{alighting}$ is the average alighting time per passenger, and

T_{ab} is the interaction between boarding and alighting.

2.3 Bus Schedule Adherence

Time Point (TP) adherence is defined as follows.

$$t_{adherence} = t_{departure} - t_{schedule} \quad (2\text{-}9)$$

Where,

$t_{adherence}$ is the schedule adherence, $t_{adherence} > 0, late;\ t_{adherence} < 0, early$

$t_{departure}$ is the actual departure time, and

$t_{schedule}$ is the schedule time at TP.

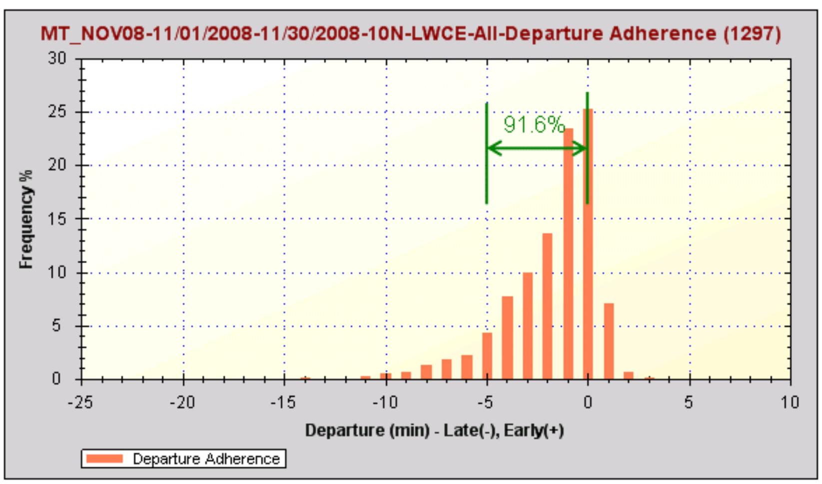

Figure 2.2. TP Schedule Adherence at LWCE NB

For example, the NB node LWCE has an On-Time Performance (OTP) of 91.6%, as plotted in Figure 2.2, when using the 5-min later and 1-min early criteria. Currently, Metro Transit uses 3-minute late threshold to trigger TSP requests.

2.4 Priority Acknowledgement Rules

After receiving a signal priority request from a bus, the signal controller has to determine whether or not to grant the request. Only one bus will get the priority service if there are multiple requests at the same intersection from buses on different approaches. The signal controller will ignore all bus priority requests if there is an emergency vehicle preemption request. The signal controller will consider the following three components when determining which bus will receive the priority service.

2.4.1 *Priority Request Time, Time Factor (TF)*

$$TF(A,B)=\begin{cases} A=W_T, B=1 & t_A<t_B \\ A=1, B=W_T & t_A>t_B \end{cases} \tag{2-10}$$

Bus *A* will receive priority if it requests earlier than bus *B* does, where W_T is the request time weighting factor ($W_T \geq 1$).

2.4.2 *Bus Schedule Adherence, Lateness Factor (LF)*

$$LF = W_L \times T_{Late} \tag{2-11}$$

Where W_L is the bus late time weighting factor ($W_L \geq 1$) and T_{Late} is the number of minute the bus was late. $LF = 0$ when bus is ahead of its schedule.

2.4.3 *Number of Passengers, Passenger Factor (PF)*

$$PF = W_P \times N_{passenger} \tag{2-12}$$

Where W_P is the bus passenger count weighting factor ($W_P \geq 1$) and $N_{passenger}$ is the number of passengers on the bus.

The priority acknowledgement functions for bus *A* and *B* are defined as follows.

$$\begin{aligned} f(A) &= TF(A,B)\times\{LF(A)+PF(A)\} \\ f(B) &= TF(A,B)\times\{LF(B)+PF(B)\} \end{aligned} \tag{2-13}$$

If the priority acknowledgement function, *f(A)* is greater than *f(B)*, bus A will be granted signal priority. No signal priority request is granted if the acknowledge function *f* equals zero, which means there are no passengers on the bus and no delay on bus schedule adherence.

2.5 Signal Timing Treatment

The projected signal phase estimated arrival time for a bus passing a signalized intersection can be calculated using the equations discussed in the previous section. When the projected signal phase coincides with the priority phase, which is the phase where a bus requires passing through an intersection, green extension is considered if the remaining green time is insufficient. However, if the projected arriving phase is different from the priority phase, phase arrangement, such as phase suppression or red truncation, is needed to provide green time to the buses. A minimum green time has to be served prior to terminating the phase.

There has been some concern about returning the intersection timing to its original coordination after providing signal priority to buses. Some priority strategies require many cycles before the signal timing is resynchronized to its regional coordination (Siemens 2002). Recently, an advanced controller provides the signal priority recovery with a cycle by including optional transit phases in the timing plan (Siemens). The bus signal priority strategy will resynchronize to its neighbor intersections in the next cycle by reducing the amount of green time extended in the next cycle priority phase. Signal priority requests in the following cycle will be ignored in order to facilitate coordination recovery. For example, if the request from bus *A* or *B* in cycle *i* was granted at an intersection, priority requests from bus *C* and *D* will not be considered because cycle *i+1* will be used for coordination recovery.

2.6 Signal Priority Implementation

The priority strategy was implemented using the C programming language and compiled on a Linux host machine. Executable binary code was then downloaded to each Onboard Unit (OBE). When a bus travels within the wireless communication range, the signal priority program will continuously monitor the bus location and speed. Bus location and its distance corresponding to the next bus stop and signalized intersection were calculated to identify a nearside versus a far-side bus stop scenario. The control diagram for the priority strategy is shown in Figure 2.3. Bus dwell time at each stop was computed based on the passenger arrival using the Poisson distribution. Bus travel times to the intersection and the bus stop were calculated to determine when to submit priority request prior to its arrival at the signalized intersection.

Signal priority settings in the controller were programmed to provide green extension or red truncation. The traffic signal controller is running on a background coordination cycle to ensure that the intersection returns back to its timing and stays coordinated with the neighboring intersections.

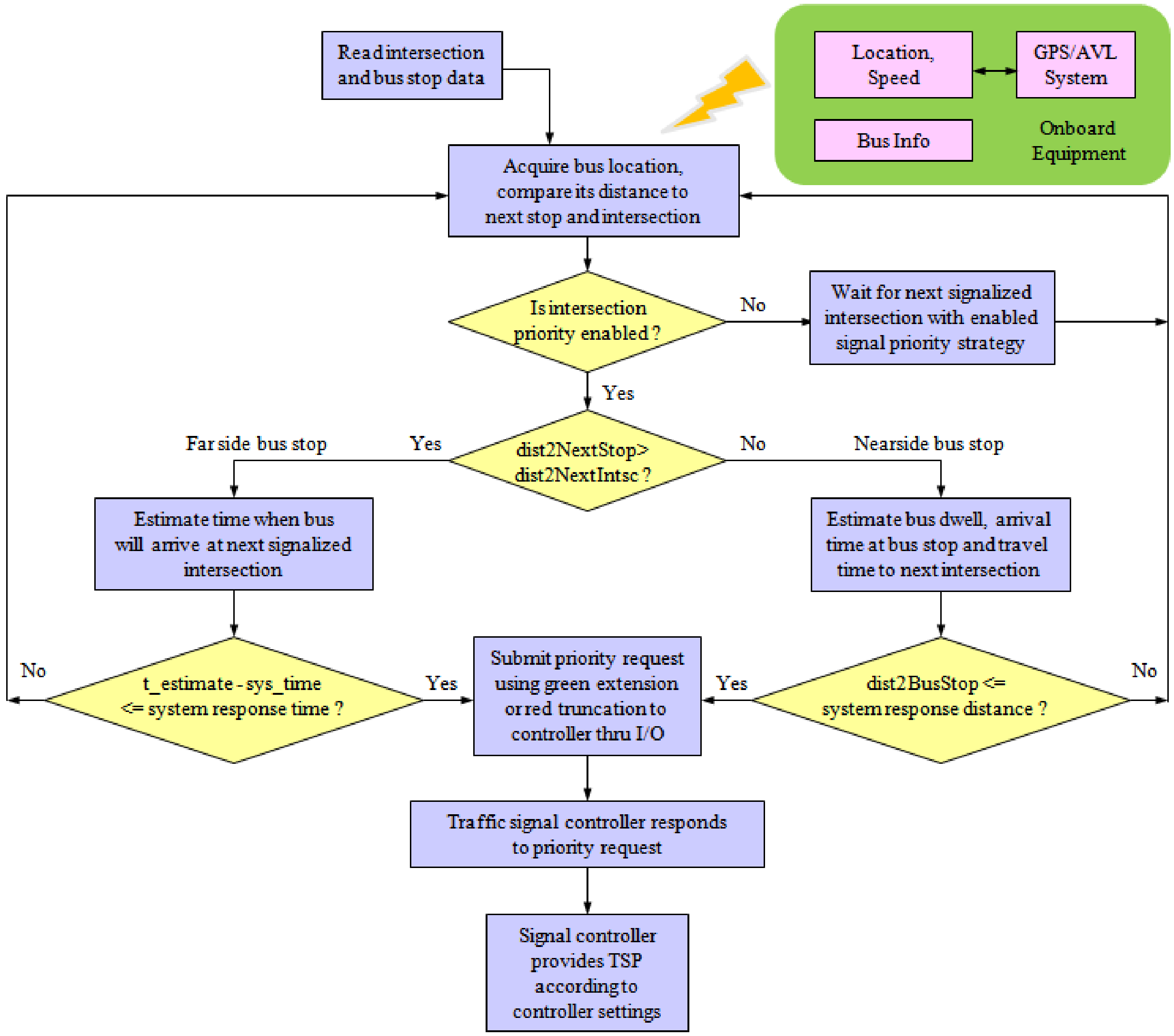

Figure 2.3. Block Diagram of Transit Signal Priority Strategy

2.7 Timepoint Model

Timepoint based bus service regression models are developed using ADCS data from Metro Transit and traffic data from the City of Minneapolis. Numbers of passengers boarding, alighting and on-vehicle standees are the primary factors contributing to the dwell time at stops. Several dwell time models can be found in the literature (Ficker 2011) using individual transit route as case study (as listed in Table 2.1). These studies did not include information about stop characteristics, bus type, and fare payment type. In additional to considering primary factors contributing to dwell time model, Milkovits (2008) considered secondary factors such as crowding, fare payment type and bus type in dwell time modeling on Chicago Transit Authority (CTA) buses. Milkovits concluded that smartcard fare payment has 1.5 sec shorter transaction time than the magnetic stripe tickets when bus is not crowded. Historical performance and parameters obtained from TP model are used for bus dwell time estimation at each stop.

Table 2.1. Dwell Time Studies Using Primary Factors

Reference	Dwell Time Model	Notes	Location
Feder (1973)	DT = 1.31 + 2.573*(Na + Nb) + ε		
Levinson (1983)	DT = 5.0 + 2.75*(Na + Nb) + ε		Several cities in U.S.
Guenthner & Sinha (1983)	DT = 5.0 - 1.2*ln(Na + Nb) + ε	Negative binomial	Milwaukee, Wisconsin
Guenthner & Hamat (1988)	DT = 2.25 + 1.81*Na, or DT = -0.27 + 5.66*Nb + ε	Separate boarding, alighting model	Southeastern Michigan Transportation Authority (SEMTA)
Lin & Wilson (1992)	DT = 9.24 + 0.71*Nb + 0.52*Na + 0.16*Ns + ε	Ns_d: # of departing standees	Massachusetts Bay Transportation Authority (MBTA)
Puong (2000)	DT = 12.22 + 2.27*Nb_d + 1.82*Na_d + 0.00062*Ns_d*Nb_d + ε	Nb_d: # of boardings/door Na_d: # of alightings/door Ns_d: # of departing standees	Massachusetts Bay Transportation Authority (MBTA)
Bertini & El-Geneidy (2004)	DT = 5.8 + 0.85*Na + 3.6*Nb + ε	R^2 = 0.47	TriMet, Portland, Oregon
Dueker et al. (2004)	DT = 5.136 + 3.481*Nb - 0.04 Nb^2 + 1.701*Na - 0.031*Na^2 - 0.144*Ontime + 1.364*TOD + ε	Ontime: dummy variable TOD: Time of day dwell effects, 1.364 sec	TriMet, Portland, Oregon
Milkovits (2008)	DT = -0.48 + Max(3.66*Nb + 2.26*Na_f, 2.7*Na_r) + 0.0013*Crowding + ε	Na_f: # of alighting by the front door Na_r: # of alighting by the rear door	RTE 63, Chicago Transit Authority (CTA), Chicago, IL
Ficker (2011)	DT = 5.044 + 0.455*Ns + 1.022*Na_f + 2.553*Nb + ε	Ns: # of standing passengers Na_f: # of alighting by the front door	Purdue University campus shuttle

Where, Na: # of alighting passengers, Nb: # of boarding passengers, ε: unmodeled error

A TP model describes the bus dwell and delay time between the check-in and check-out boundaries of a TP zone. TP time model incorporates parameters such as number of passengers boarding and alighting, bus type, fare payment type, and stop location characteristics. TP time can be modeled by analyzing the time distribution for near vs. far-side stops and early vs. late buses. An application developed previously by Liao & Liu (2010) allows users to visualize the histogram of the early arrival bus dwell time at selected TP. The late bus TP time distribution should include minimum possible holding assuming a bus operator will be less likely to hold at a TP when the bus is behind schedule. However, bus delay due to traffic light is hidden in the distribution. A bus TP time model can be formulated by including hour of day, number of board and alighting, seat availability, electronic fare transactions, near side or far side bus stop, and bus type parameters as described in equation (2-14).

$$TP\ Time\ Model = \mathcal{F}\{HourOfDay,\ StopAttribute,\ BusAttributes,\ PaymentType\} \quad (2\text{-}14)$$

Where,

$StopAttribute = \mathcal{F}\{Near/Far\ Side, Traffic\ Signal\ (Cycle, Split, Phase), Geometry\}$,
$BusAttribute = \mathcal{F}\{SeatAvailability, Board, Alight, LowFloor, Articulate\}$, and
$PaymentType = \mathcal{F}\{GoTo\ Card, Cash, Others\}$.

Most of required parameters can be obtained or indirectly derived from the AVL/APC database except the electronic fare transition records. The automatic fare collection system records the transaction time, type and location. A data mining methodology was developed to match electronic fare transaction data to AVL/APC database and extract the fare transaction counts at stop level for each APC-equipped bus. The TP time model was developed as described in equation (2-15). TP time data of all late/on-time buses without wheelchair lift are included. Timepoint OSMN and 73CE were also excluded because both were located on a branch and no available traffic information. The total data size (N) is 9,813 from the Nov. 08 dataset.

TP time regression model:
tp_time_sec ~ (no_activity + board + alight_only + goto_cnt + FS + h10_16 + h17_19 + h20_21 + dir_North + cyc_fixed_flag + split_sq_cyc_fixed + LT + RT) (2-15)

Where,

tp_time_sec:	Timepoint time in seconds
no_activity:	Dummy variable
alight_only:	Dummy variable
board:	Number of boardings
goto_cnt:	Number of GoTo card users
FS:	Dummy variable, farside stop
h10_16:	Dummy variable, 10AM-4PM
h17_19:	Dummy variable, 5PM-7PM
h20_21:	Dummy variable, 8PM-9PM
dir_North:	Dummy variable, northbound bus
cyc_fixed_flag:	Dummy variable, fixed time signal
split_sq_cyc_fixed:	Green split, split_sq_cyc_fixed = $(\text{green time})^2/(\text{cycle time})$
LT:	Dummy variable, left turn
RT:	Dummy variable, right turn

The regression results of the refined RTE10 TP time model using equation (2-15) were listed in Table 2.2. The model has an intercept of 30.45 sec. There is 15.27 sec of TP time savings when there is no service activity. Each boarding will add about 5.58 sec to the TP time which is similar to the value from the preliminary model. Each alight adds 1.47 sec to TP time. A FS stop will deduct 2.57 sec from the TP time. Each GoTo card user will save 3.69 sec. i.e., 1.89 sec per GoTo user boarding. Outbound (NB) direction has 4.85 sec less TP time than inbound (SB) traffic. Left and right turning at a TP contribute additional 30.34 sec and 26.6 sec to the total TP time, respectively. The R^2 value of the model is 0.54 which indicated that 54% of the data variability can be explained by the refined regression model.

Table 2.2. TP Time Regression Model (Adjusted R-squared: 0.5422)

Coefficients	Estimate	t value
(Intercept)	30.45	43.58
no_activity	-15.27	-26.82
board	5.58	28.85
goto_cnt	-3.69	-6.04
alight	1.47	7.55
alight_only	-2.10	-9.29
FS	-2.57	-5.42
h10_16	4.66	8.98
h17_19	6.57	10.47
h20_21	4.81	6.11
dir_North	-4.85	-10.79
cyc_fixed_flag	0.31	21.10
split_sq_cyc_fixed	-0.16	-2.93
LT	30.34	26.46
RT	26.60	24.29

3 UMN TRANSIT SIGNAL PRIORITY SYSTEM

Commercial-Off-The-Shelf (COTS) embedded stand-alone Single Board Computer (SBC), as shown in Figure 3.1, was selected for the TSP system development. The TS-7800 series SBC, manufactured by the Technologic Systems (http://www.embeddedarm.com/), was programmed to interface with bus Vehicle Control Unit (VCU) and EMTRAC wireless communication radio.

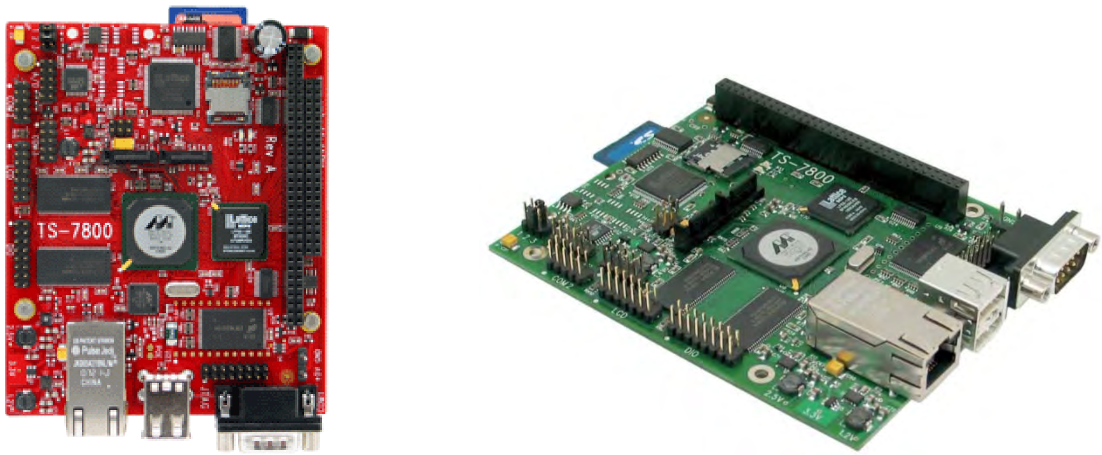

Figure 3.1. TS-7800 Single Board Computer

3.1 Embedded Computer

The TS-7800 SBC features the Marvell 500MHz ARM9 CPU, which provides high end features such as, gigabit Ethernet, dual high-speed USB 2.0 host/slave ports, and dual SATA II ports. The TS-7800 embedded computer runs on Linux 2.6 kernel with a full Debian distribution with a minimum of 128MB DDR-RAM and 512MB on-board NAND flash. It also offers a CompactFlash (CF) or Secure Digital (SD) card socket to allow removable media storage and support bootable media. Linux (www.linux.org/) kernel for each embedded computer is built on a Linux host machine which compiles and creates a bootable Linux OS image on a SD card. Detail information on TS-7800 SBC is available online at manufacturer's website, http://www.embeddedarm.com and Appendix C.

Figure 3.2. TS-7800 SBC Embedded System

3.2 System Integration and Communication Interface

System integration of the embedded TSP system and software design and development are the key elements in developing the wireless signal priority system as shown in Figure 3.3. The EMTRAC TSP onboard unit is connected to bus Vehicle Control Unit (VCU) to obtain door status and schedule adherence information. In regular operation with UMN TSP unit, the EMTRAC unit will monitor vehicle location and schedule adherence to determine when or where to request signal priority. When the UMN TSP is installed, the UMN TSP system will temporary disable the TSP algorithm running in the EMTRAC unit through a customized serial interface. EMTRAC GPS data is available to UMN TSP system and the TSP request from UMN TSP unit is sent through the serial interface to the EMTRAC wireless communication interface then to roadside unit residing in traffic controller cabinet.

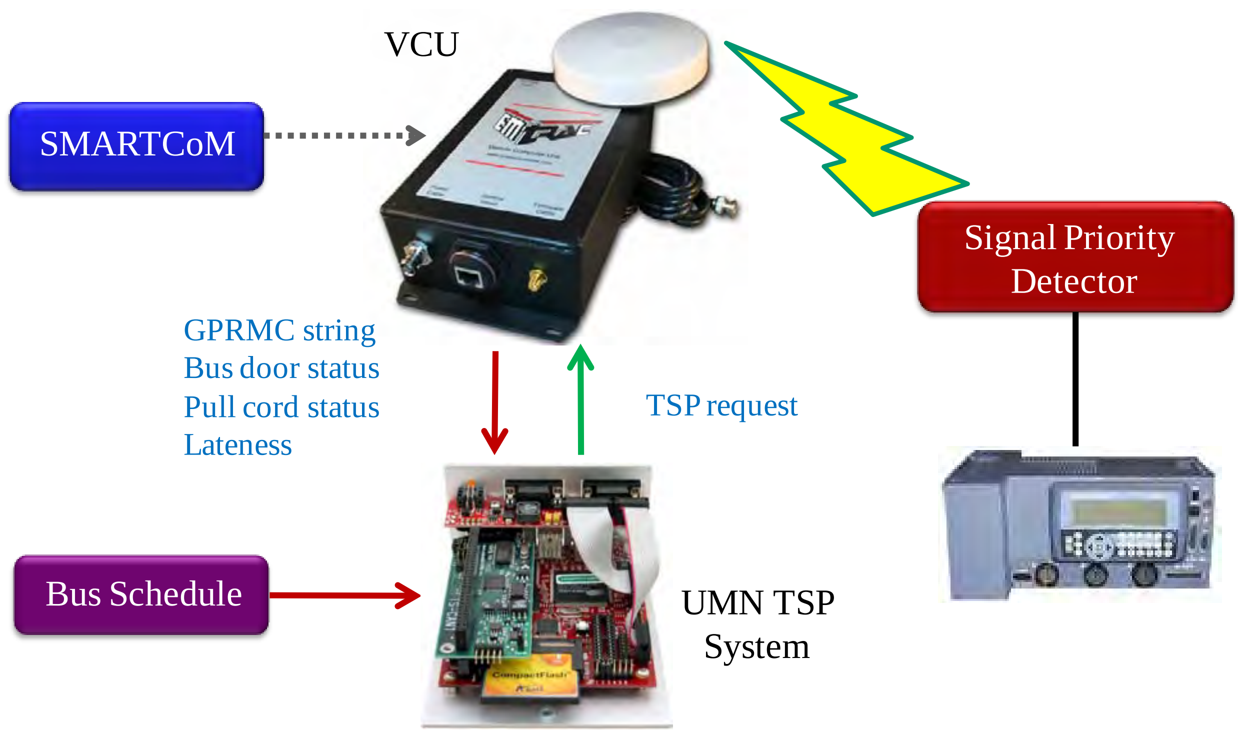

Figure 3.3. TSP System Diagram

3.3 Software Implementation

Software implementation flowchart is illustrated in Figure 3.4. The UMN TSP system will run automatically when power is on. It will establish a communication channel (COM port) to EMTRAC and then transmit commands to disable EMRAC TSP and request for GPS string continuously during the initialization. A separate process will keep reading incoming GPS data to update current location, speed, and heading. Another process will execute the TSP algorithm to monitor current location and determine next bus stop, intersection, and schedule lateness by comparing real-time data to schedule and route stop sequence information stored in the database. Signal priority request will be sent to the roadside controller through the EMTRAC wireless interface when the bus is more than 3-min late and ready to cross the intersection. Sample source codes are listed in Appendix F.

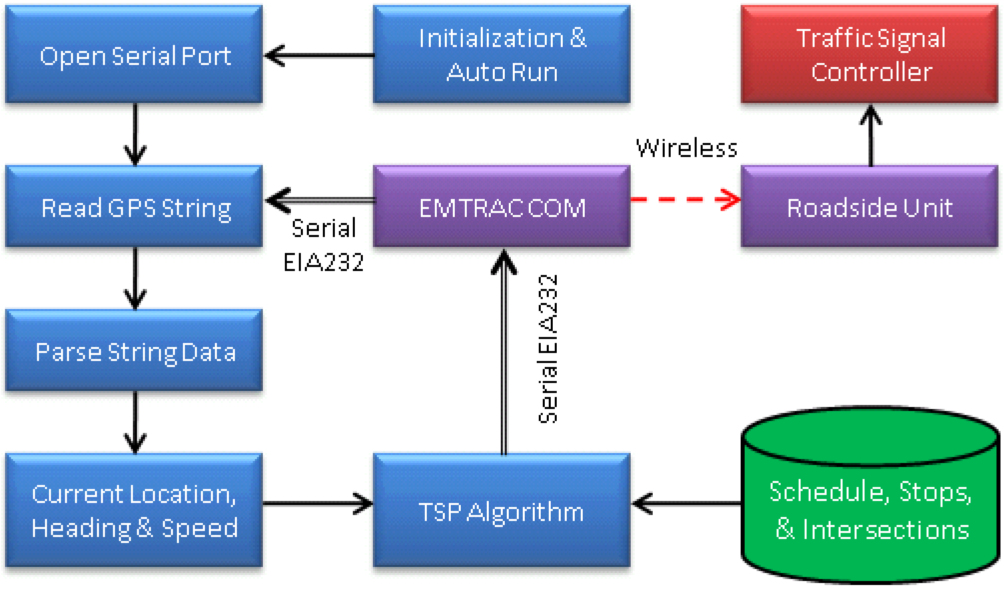

Figure 3.4. Software Implementation Flowcharts

4 FIELD EXPERIMENTS

As part of the Urban Partnership Agreement (MnDOT, 2008), Metro Transit, MnDOT, and the City of Minneapolis have implemented Transit Signal Priority (TSP) along Central Avenue (bus route 10/59) in parallel of I-35W from north of Minneapolis to south of I-694. Transit performance before and after the deployment of a TSP strategy can be examined through the data analysis process to evaluate the effectiveness and benefit of a TSP strategy. Bus schedules can potentially be adjusted to take advantage of a TSP strategy through running time and timepoint time analysis.

4.1 Test Site

Bus route 10, as shown in Figure 4.1, was selected for performance analysis and modeling in this study. Bus line 59 operates on the same route with RTE10 but offers limited-stop express service. The blue segments on Central Avenue are local streets and red segment is a highway (TH-47). There are 27 signalized intersections and 39 bus stops between 2nd street SE and 53rd Avenue NE on Central Avenue (about 5.5 miles). Signal controllers south of 27th Avenue are managed by the City of Minneapolis and the signal controllers north of 27th Avenue are managed by MnDOT. All 27 signalized intersections are equipped with wireless TSP roadside equipments. As part of the UPA funding, over 700 Metro Transit buses are instrumented with TSP onboard systems. Prior to TSP implementation, the signals south of 27th Avenue (inside City of Minneapolis jurisdiction) are previously pre-timed.

According to the City of Minneapolis traffic engineer, all intersections on Central Avenue between 2nd and 27th street are changed to semi-actuated, except Broadway and Lowry. Broadway has actuated left-turn phases. Lowry is pre-timed, including the left-turn arrows. Semi-actuation controllers were installed around July 2009. The semi-actuated signal controllers along Central Avenue in Minneapolis has the following cycle settings; off-peak = 70 seconds, AM peak = 100 seconds, PM peak = 110 seconds, and Midday = 90 seconds. Prior to semi-actuation installation, fixed-time signals were configured with the following cycle time; AM/PM peak = 90 seconds and off-peak = 70 seconds. Intersections north of 27th street, managed by MnDOT, are already semi-actuated prior to TSP implementation. Complete listing of signalized intersections and bus stops are included in Appendix D.

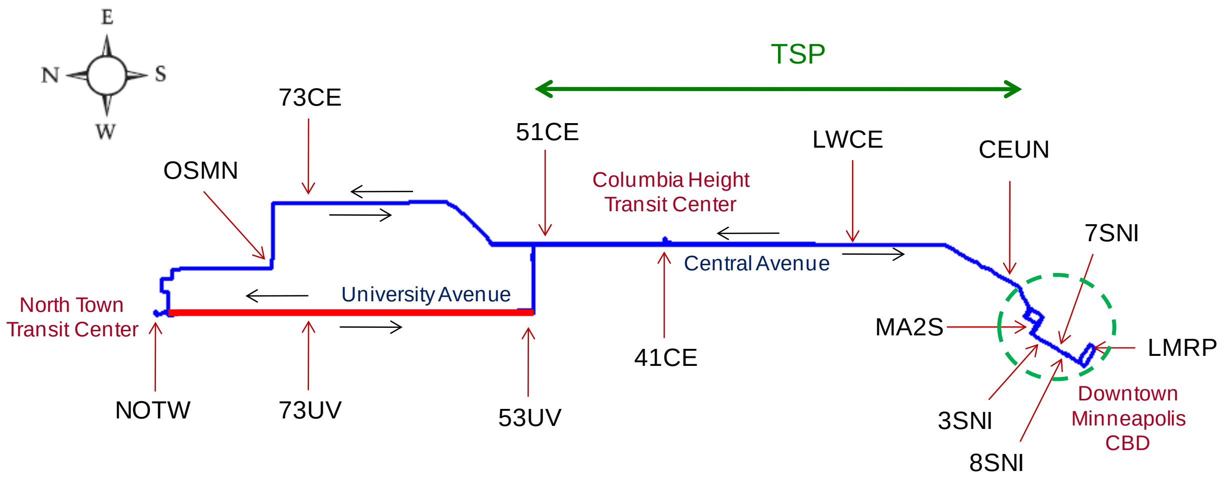

Figure 4.1. Timepoints of Metro Transit Bus Route 10/59

4.2 Conditional Priority Request

In addition to the signal priority acknowledge rules discussed in section 2.4, the following two rules are imposed on the Minneapolis TSP system to limit number of TSP requests and minimize traffic delays on side streets.

Headway Limitation: is the minimum time duration between two consecutive TSP requests. Traffic signal controller will not respond to TSP request if consecutive TSP requests were sent less than the headway limitation.
Lateness Threshold: is the minimum bus lateness required to request for TSP. The signal priority request will only be considered if a bus is late more the lateness threshold.

The TSP activation is initially configured to 10-minute for both headway limitation and lateness threshold in Minneapolis (2^{nd} St. to 37^{th} Ave.) in Feb. 2010 by the City of Minneapolis. Later in June 2010, the lateness threshold was changed to 5-minute for all TSP-equipped signals. The lateness threshold was further reduced to 3 minute and headway limit was changed to 8-minute for all signals after Nov. 2010.

4.3 Data Collection

AVL/APC data of 42 RTE 10 buses operating during the test period (3/28/2011 - 4/8/2011) were collected. Bus performance of 4 vehicles (Veh_ID: 7212~7215) equipped with UMN TSP system were analyzed and compared with the other buses during the test period. Available data types collected by AVL/APC system are listed in Table 4.1. Over the two week test period, 1569 bus trips were operated and over 100,000 records of data from 42 buses were collected. Available data types collected from UMN and EMTRAC TSP systems are listed in Tables 4.2 and 4.3, respectively.

Table 4.1. Sample Data Collected from Metro Transit ADCS System

Data Name	Sample Data	Data Name	Sample Data
calendar_id	120110318	apc_msg_time	NULL
calendar_date	3/18/2011	act_dep_at_tp	16194
veh_id	7211	confidence	NULL
time_bracket_start	16140	trip_level_accuracy	95
time_bracket_end	18060	seg_from	51CE
block_number	1219	seg_to	51CE
trip_number	1	run_number	2007
service_id	WK	odometer	854
line_id	10	act_dist_from_sched_ft	NULL
line_direction	South	flg_badtrip	NULL
line_direction_number	1	flg_apc_fixed_at_common_tp	NULL
pat_id	51CELMRP00	flg_apc_count_was_interleaved	NULL
stop_sequence_number	1	flg_bad_bus	1
node_id	51CE	xmit_msg_id	NULL
site_id	17142	sched_tp_adhere_vl_percent	98
site_latitude	45.06119537	oper_id	1586
site_longitude	-93.2477951	route_vers	NULL
board	3	flg_nonreprsvc	NULL
alight	1	c_load	16
sched_time	16140	flg_picklevel_outlier	18
act_arr_at_tp	15736	flg_maxload	NULL

Table 4.2. Sample Data Collected from UMN TSP Onboard System

Data Name	Sample Data
Timestamp	15:55:23
Direction	N
X (m)	859192.5781
Y (m)	320944.4274
Latitude (Deg)	44.986015
Longitude (Deg)	-93.249412
Trip Number	242
Adherence (min)	-6
Next Stop ID	14954
Next Stop Index	16
Distance to Next Stop (m)	420.526745
Next Intersection ID	26
Distance to Next Intersection (m)	441.577079
TSP Flag	-1

Table 4.3. Sample Data Collected from EMTRAC TSP System

Data Name	Sample Data	Data Name	Sample Data
INDEX	127817	VELOCITY	19
OPERATION_INDEX	785	HEADING	2
LOG_INDEX	778	START_DATE	2011-03-29 17:47:26
INT_ID	24	STOP_DATE	2011-03-29 17:47:46
ZONE_ID	24	DURATION	0
DIR	N	STATUS	32
VEH_ID	533	TSP_STATUS	True
PRIORITY	2	call_duration_sec	20

Prior to deploying UMN TSP units on Metro Transit buses, an EMTRAC VCU was installed on a passenger vehicle to test the communication interface and TSP algorithm developed on the UMN TSP system. Numerous tests were conducted by driving the passenger vehicle along Central Avenue to verify and validate priority request, GPS reception, etc. Additional tests on wireless TSP communication with Roadside Unit (RSU) were also conducted at the traffic laboratory in the City of Minneapolis Public Works department.

5 RESULTS AND DATA ANALYSIS

In 2009 and 2010, Transit Signal Priority (TSP) was instrumented on Metro Transit buses and installed at all signalized intersections between node CEUN and 51CE along the Central Avenue. Four separate months of dataset that covers the implementation and operation periods of TSP were analyzed to evaluate the existing TSP performance as discussed in section 5.1.

Data of bus route 10 operating during the two weeks of test period (3/28/2011 - 4/8/2011) were analyzed. Performances of four buses using UMS TSP algorithm were compared with the other RTE10 buses operating during the same test period. TP time, inter-TP link Travel Time (TT), and trip TT inside the TSP-enabled links (between CEUN and 51CE nodes, about 5.5 miles) are discussed in section 5.2.

5.1 Existing TSP System Evaluation

5.1.1 Scheduled Link Travel Time

Scheduled link travel time for segments north of timepoint 41CE remains almost the same between Nov. 08 and Oct.10 for both NB and SB directions as shown on the right of Figure 5.1 and 5-2. In average, the scheduled travel time between timepoint CEUN and LWCE was adjusted almost 2 minutes (1.85 min in NB and 1.49 min in SB) shorter in Oct. 10 than the schedule in Nov. 08 in both directions. Similarly, the average scheduled travel time between timepoint LCWE and 41CE was about 1 minute (1.11 min in NB and 1.16 in SB) shorter in Oct. 10 as compared to that in Nov. 2008, as illustrated on the left two groups of bar charts illustrated in Figures 5.1 and 5.2, respectively.

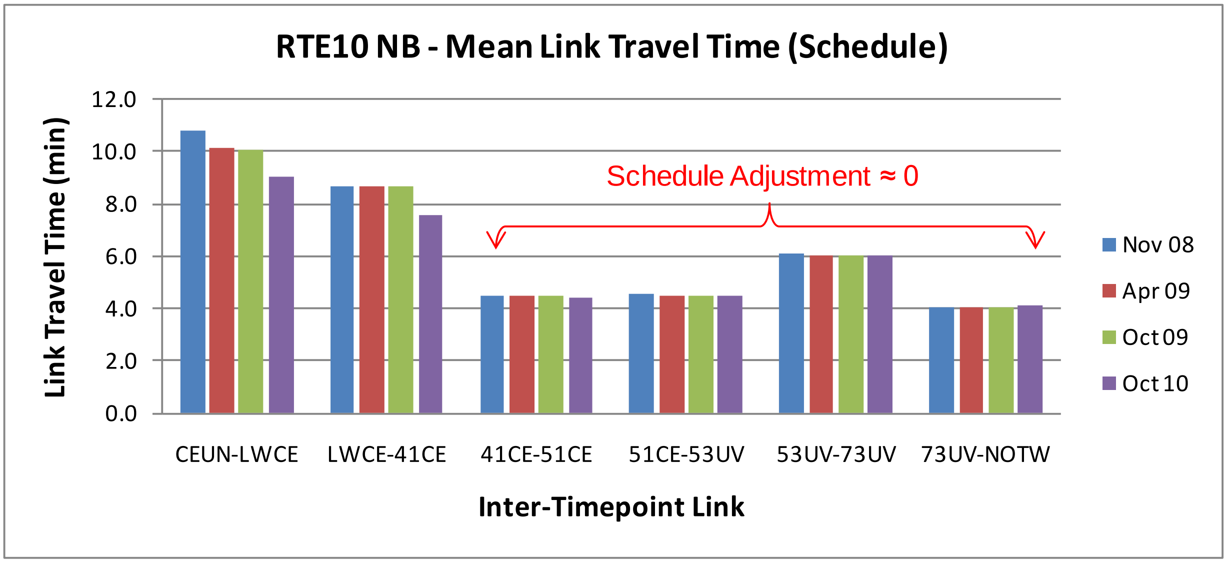

Figure 5.1. Route 10 NB Scheduled Link Travel Time

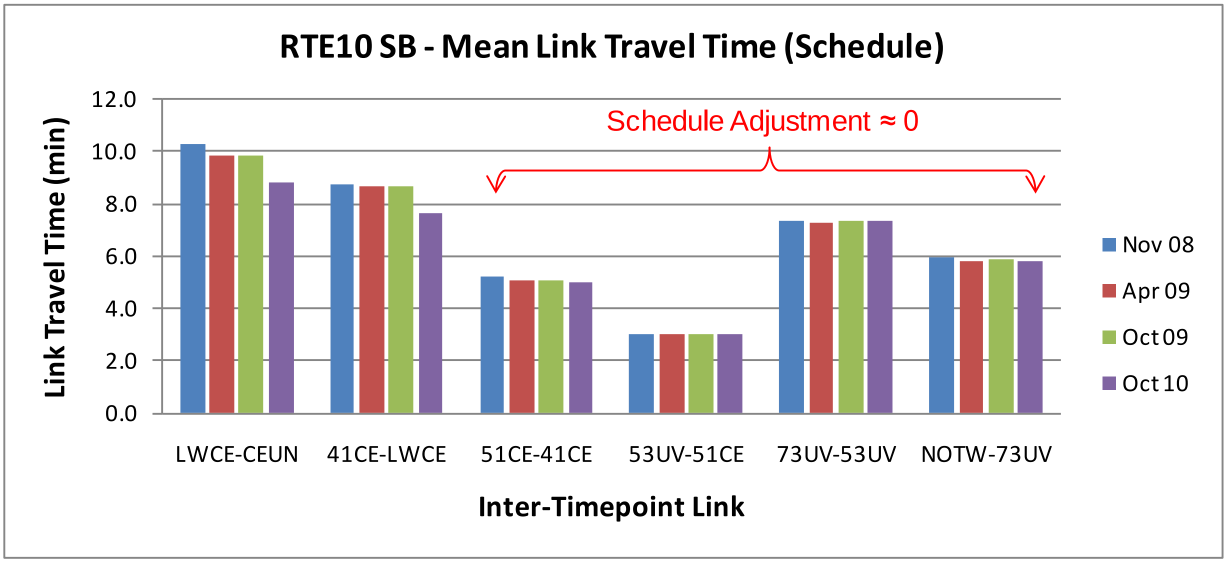

Figure 5.2. Route 10 SB Scheduled Link Travel Time

5.1.2 Observed Link Travel Time

Intersections between CEUN and 51CE were equipped with TSP capability. Trip travel time within the CEUN-51CE segment was analyzed as shown in Figure 5.3. Scheduled versus observed travel time statistics for each dataset in both directions were listed in Table 5.1. Travel time of NB segment in Oct. 10 decreases by about 1.4 minutes (6%) while the schedule travel time reduces by about 3 minutes (13%) as compared to that in Nov. 08 as illustrated in Table 5.2. Travel time of SB segment in Oct. 10 decreases by about 1 minute (4%) while the schedule travel time reduces by about 1.3 minutes (5%) as compared to that in Nov. 08. Lateness threshold for TSP request was further reduced from 5-min to 3-min threshold by the City of Minneapolis and Metro Transit in Nov. 10. Existing TSP system improves the bus travel time by about 4-6% after reducing the scheduled travel by about two minutes.

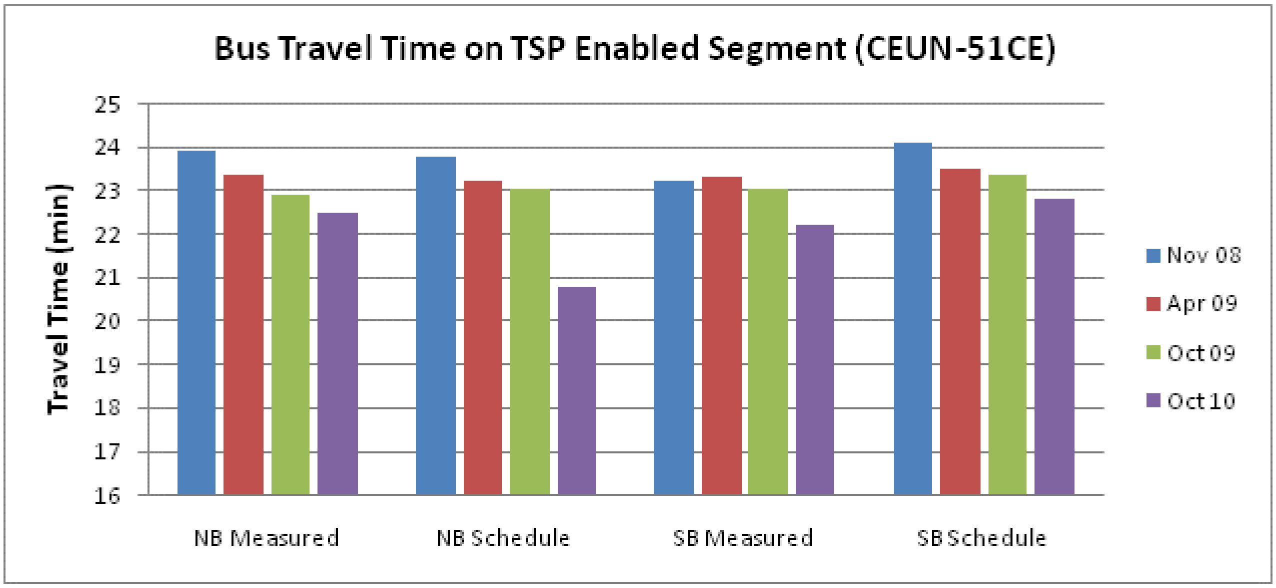

Figure 5.3. RTE10 Travel Time (CEUN-51CE) Comparisons (TSP-Enabled Segment)

Table 5.1. Travel Time Statistics of TSP Enabled Links

Trip Travel Time		Nov 08		Apr 09		Oct 09		Oct 10	
Trip	Data Type	Mean	Stdev	Mean	Stdev	Mean	Stdev	Mean	Stdev
Northbound CEUN-51CE	Measured	23.90	3.18	23.38	4.02	22.92	3.08	22.48	4.13
	Schedule	23.79	2.03	23.24	1.77	23.06	1.88	20.79	1.95
Southbound 51CE-CEUN	Measured	23.25	3.34	23.34	3.81	23.04	2.59	22.24	2.58
	Schedule	24.11	2.05	23.50	1.76	23.38	1.85	22.80	1.98

Table 5.2. RTE10 Travel Time Reduction

TT Reduction (Oct 10 - Nov 08)/Nov 08		
Northbound CEUN-51CE	Measured	-6%
	Schedule	-13%
Southbound 51CE-CEUN	Measured	-4%
	Schedule	-5%

5.1.3 TSP Call Duration

Current TSP system defines the call duration as the time from the initiation of TSP request to the time a bus left intersection virtual zone. Histogram of TSP request duration was plotted in Figure 5.4. Over 75% of the call durations were less than 15 seconds. Current TSP roadside equipment does not record if TSP request was granted or not by the traffic signal controller.

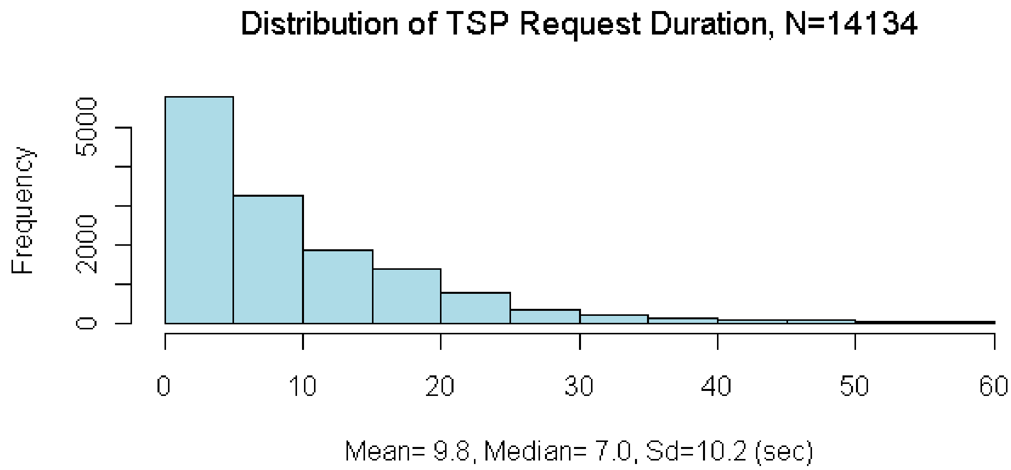

Figure 5.4. Distribution of TSP Request Duration

5.2 UMN TSP Comparisons

Performances of 4 test buses (equipped with UMN TSP system) were compared with the other RTE 10 buses running regular TSP algorithm.

5.2.1 Time Point (TP) Time Analysis

A TP node is a control point for evaluating schedule adherence as illustrated in Figure 5.5. A TP is a zone defined virtually around a bus stop, for example, 60 meter (200 ft) upstream and downstream from a bus stop. TPs are placed throughout the entire transit network to systematically monitor schedule adherence and running time performance using the AVL system.

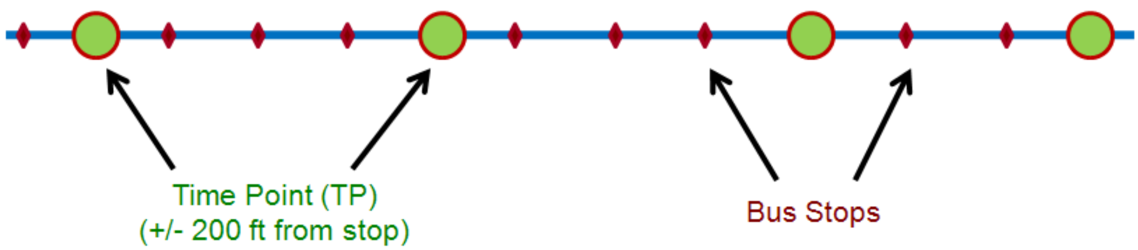

Figure 5.5. Time Point and Bus Stop

Table 5.3 listed the TP time comparisons between the 4 buses running UMN TSP algorithm and all other RTE 10 buses in the same test period. In general, the TP time of the UMN TSP buses in SB is significantly lower than the TP time of all other buses. In NB, difference between the TP time of test and regular buses are not significant (0.6 - 4.2 sec). In fact, the TP time of test buses at node LWCE & 41CE were actually 2.4 sec longer than the other RTE 10 buses.

Table 5.3. Time Point Time Comparisons

Time Point Time (min)		4 Buses w/ UMN TSP			All other Buses			Mean Diff (min)	Mean Diff (%)
Direction	Time Point	Mean	STDEV	N	Mean	STDEV	N		
NB	CEUN	0.91	0.56	55	0.92	0.48	757	-0.01	-0.9%
	LWCE	1.27	0.70	55	1.24	0.68	755	0.04	3.0%
	41CE	0.23	0.33	54	0.19	0.33	753	0.04	23.9%
	51CE	0.32	0.30	45	0.39	0.58	601	-0.07	-18.0%
SB	51CE	0.30	0.49	41	0.48	1.48	533	-0.18	-37.4%
	41CE	2.09	4.34	51	2.63	4.73	673	-0.54	-20.7%
	LWCE	1.31	0.51	51	1.50	0.92	696	-0.19	-12.5%

5.2.2 Link Travel Time Analysis

Comparisons of inter-TP link travel time between the test buses and all other RTE 10 buses during the two-week test period were listed in Table 5.4 as follows. Overall, the link TT of the four test buses are about 30-36 seconds shorter than the TT of all other RTE 10 buses except at link 51CE-41CE in SB.

Table 5.4. Link Travel Time Comparisons

Link Travel Time (min)		4 Buses w/ UMN TSP			All other Buses			Mean Diff (min)	Mean Diff (%)
Direction	Link	Mean	STDEV	N	Mean	STDEV	N		
NB	CEUN-LWCE	9.69	1.88	73	10.02	1.97	729	-0.33	-3.3%
	LWCE-41CE	8.57	2.20	73	9.14	1.96	725	-0.57	-6.2%
	41CE-51CE	4.09	1.25	60	4.69	1.56	592	-0.60	-12.7%
SB	51CE-41CE	5.74	2.35	54	5.36	1.98	531	0.38	7.1%
	41CE-LWCE	7.75	2.72	58	8.24	3.20	584	-0.49	-5.9%
	LWCE-CEUN	10.03	1.73	60	11.10	2.30	662	-1.07	-9.7%

5.2.3 Trip Travel Time on TSP-Enabled Segment

Trip travel time between node CEUN and 51CE were analyzed and compared as listed in Table 5.5. The average NB travel time of the four test buses is about 1.5 min (6.4%) shorter than that of the other buses. The average test bus travel time in SB is about 0.6 min (2.6%) shorter than the TT of all other RTE10 buses. The TT of test buses in NB is slightly shorter (22.79-22.12=0.67 min) than the TT in SB. The scheduled trip travel time on this TSP enabled segment is listed in Table 5.6. The average scheduled travel time in NB is 1.54 min shorter than the schedule in SB. Because of the 3-minute lateness of conditional TSP rule and the shorter schedule time in NB, the buses are more likely to being late and therefore receiving more signal priority. This explains why the NB trips receive more TT reduction as compared to the TT in SB as shown in Table 5.2 and 5-5.

Table 5.5. Trip Travel Time Comparisons

TSP Trip TT (min)		4 Buses w/ UMN TSP			All other Buses			Mean Diff (min)	Mean Diff (%)
Direction	Trip Segment	Mean	STDEV	N	Mean	STDEV	N		
NB	CEUN-51CE	22.12	2.45	63	23.63	2.87	691	-1.51	-6.4%
SB	51CE-CEUN	22.79	2.48	57	23.40	2.91	602	-0.61	-2.6%

Table 5.6. Scheduled Trip Travel Time

Scheduled TT (min)		All RTE 10 Buses		
Direction	Trip Segment	Mean	STDEV	N
NB	CEUN-51CE	20.64	2.30	247
SB	51CE-CEUN	22.18	2.05	639

6 SUMMARY AND DISCUSSION

This project was postponed due to the delay of the UPA TSP implementation on Central Avenue. Our experiments were conducted after the Minneapolis TSP system was fully operational and stable. The objective of this project is to deploy and validate a wireless-based TSP strategy developed from earlier study (Liao & Davis, 2008) by considering the bus schedule adherence, location and speed. A stand-alone Single Board Computer (SBC) was selected for the embedded programming and system development. The TS-7800 series SBC, a commercial off-the-shelf product, was programmed to interface with bus Vehicle Control Unit (VCU) and EMTRAC wireless communication radio and bypass the EMTRAC TSP algorithm. The goal is to validate our TSP algorithm by taking advantage of the already instrumented onboard and roadside infrastructures for buses to communicate with intersection signal controllers when they are ready to request for priority.

Signal priority request output on the roadside equipment was connected to a low priority input channel on the signal controller through wirings in the controller cabinet. Program of the traffic controller was also configured and activated to accept external priority inputs. The traffic signal controller was programmed by the City of Minneapolis traffic engineer to specify corresponding delay, dwell, maximum call and extension time.

Based on our analysis results, the UPA TSP implementation reduces average bus travel time by about 4-6% on the TSP-enabled segments after two-minute reduction of RTE10 schedule. Field experiments were performed by installing four UMN TSP units on four RTE10 buses for two weeks. The EMTRAC algorithm was temporary disabled on the test vehicles. Performances of four buses using UMS TSP algorithm were compared with the other RTE10 buses operating during the same test period. TP time, inter-TP link Travel Time (TT), and trip TT inside the TSP-enabled links (between CEUN and 51CE nodes) were analyzed. The results indicated the UMN TSP algorithm averagely provides additional 3-6% of travel time reduction to RTE 10 buses. Our streets and highways are getting more and more congested as population grows and more cars enter the transportation system. We hope that providing signal priority to buses can help improve quality and reliability of transit services.

A methodological data analysis framework, as shown in Figure 6.1, was previously developed to process massive transit data including vehicle location, passenger count and electronic fare transactions (Liao & Liu, 2010). The developed data analysis framework can assist a number of research and applications such as transit route performance measurement, TSP and decision-making support for transit planning and operation. The vision is to use the data processing framework to generate deployment suggestions for TSP deployment. We plan to continue our ongoing collaboration with Metro Transit to develop tools to support transit operation, scheduling and planning.

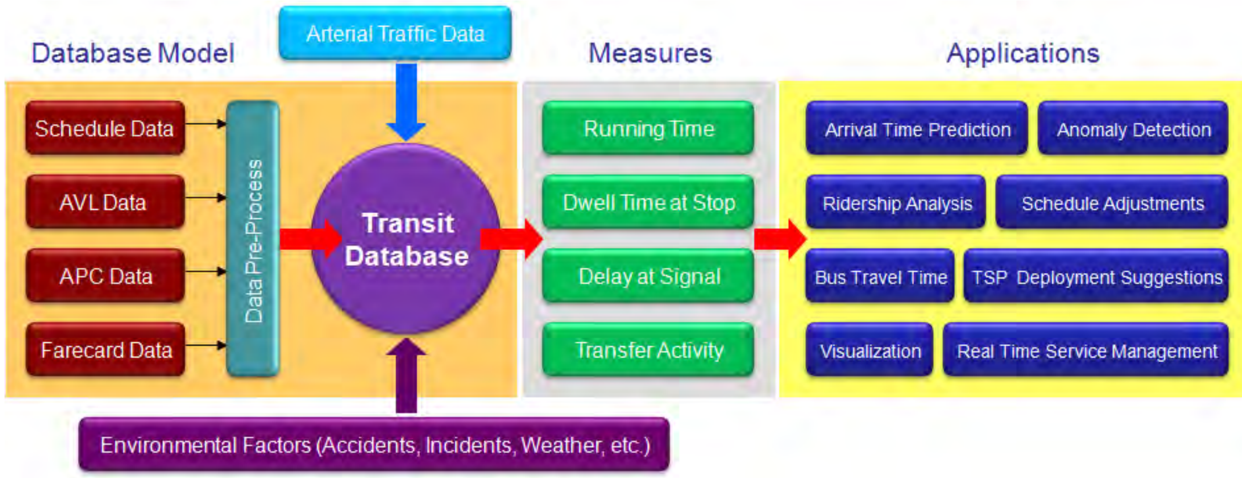

Figure 6.1. Transit Performance Data Processing Framework

REFERENCES

3M™ Opticom™ Priority Control System, http://solutions.3m.com/wps/portal/3M/en_US/Traffic_Safety/TSS/Offerings/Systems/Opticom/, accessed October 1, 2007.

Ahmed, H., El-Darieby, M., Abdulhai, B., Morgan, Y., (2008). "Bluetooth- and Wi-Fi-Based Mesh Network Platform for Traffic Monitoring." *Transportation Research Board 87th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 13-17.

Alexander, L., Cheng, P., Gorjestani, A., Menon, A., Newstrom, B., Shankwitz, C., Donath, M., (2006). "The Minnesota Mobile Intersection Surveillance System." *Proceedings of 9th International IEEE Conference on Intelligent Transportation Systems*, pp. 139-144. Toronto, Canada, September 17-20.

Altun, S.Z., Furth, P.G., (2009). "Scheduling Buses to Take Advantage of Transit Signal Priority." *Transportation Research Record*, No. 2111, pp. 50–59.

Berkow, M., Wolfe, M., Monsere, C.M., Bertini, R.L., (2008). "Using Signal System Data and Buses as Probe Vehicles to Define the Congested Regime on Arterials." *Transportation Research Board 87th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 13-17.

Bertini, R.L., El-Geneidy, A.M., (2004). "Modeling Transit Trip Time Using Archived Bus Dispatch System Data." *Journal of Transportation Engineering*, Vol. 130, No. 1, pp. 56-67.

Biswas, S., Tatchikou, R., Dion, F., (2006). "Vehicle-to-Vehicle Wireless Communication Protocols for Enhancing Highway Traffic Safety." *IEEE Communications Magazine*, Vol. 44, Issue 1, pp.74-82.

Böhm, M., Pfliegl, R., Frötscher, A., (2008). "Wireless Infrastructure-To-Vehicle Communication Technologies to Increase Road Safety along Motorways." *Transportation Research Board 87th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 13-17.

Bretherton, R.D., Hounsell, N.B., Radia, B., (1996). "Public Transport Priority in SCOOT." *Proceedings of the 3rd Annual World Congress on ITS Systems*, Orlando, FL, October 14-18.

Cathey, F.W., Dailey, D.J., (2003). "A prescription for transit arrival/departure prediction using automatic vehicle location data." *Transportation Research Part C*, No.11, pp. 241–264.

Collura, J., Rakha, H., Gifford, J., (2003). *Guidelines for the Planning and Development of Emergency Vehicle Preemption and Transit Priority Strategies*. Virginia Tech Transportation Institute, Blacksburg, VA, and George Mason University School of Public Policy, Arlington, VA.

Crout, D.T., (2005). "Evaluation of Transit Signal Priority at the Tri-County Metropolitan Transportation District of Oregon (TriMet)." *Proceedings of the 12th Annual World Congress on ITS Systems*, San Francisco, CA, November 6-10.

Dion, F., Rakha, H., (2005). "Integration of Transit Signal Priority Within Adaptive Traffic Control Systems". *Transportation Research Board 84th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 9–13.

Dueker, K.J., Kimpel, T.J., Strathman, J.G., Callas, S., (2004). "Determinants of Bus Dwell Time," *Journal of Public Transportation*, Vol. 7, No. 1, pp. 21-40. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.171.8&rep=rep1&type=pdf, accessed November 2010.

Farradyne, P.B., (2005). *VII Architecture and Functional Requirements, Version 1.1*. U.S. Department of Transportation, Washington, D.C. www.vehicle-infrastructure.org/documents/VII%20Architecture%20version %201%201 %202005_07_20.pdf, accessed November 1, 2007.

Feder, R.C., (1973). *The Effect of Bus Stop Spacing and Location on Travel Time*. Transportation Research Institute, Carnegie Mellon University, Pittsburgh PA.

Federal Highway Administration, (2004). "Traffic Analysis Toolbox Volume III: Guidelines for Applying Traffic Microsimulation Modeling Software." FHWA-HRT-04-040, Federal Highway Administration, McLean, VA.

Fricker, J.D., (2011). "Bus Dwell Time Analysis Using Onboard Video." *Transportation Research Board 90th Annual Meeting Compendium of Papers*, Washington, D.C., January 23-27.

Fitzmaurice, M., (2005). "Use of Wireless Local Area Networks in Rail and Urban Transit Environments." *Transportation Research Record*, No. 1916, pp. 42-46.

Furth, P.G., Mueller, T.H., (2000). "Conditional Bus Priority at Signalized Intersections, Better Service with Less Traffic Disruption." *Transportation Research Record*, No. 1731, pp. 23-30.

Furth, P.G., SanClemente, J.L., (2006). "Near Side, Far Side, Uphill, Downhill, Impact of Bus Stop Location on Bus Delay." *Transportation Research Record*, No. 1971, pp. 66-73.

Ghanim, M., Dion, F., Abu-Lebdeh, G., (2007). "Projected Transit Arrival Time Prediction Tool for Transit Signal Priority with Nearside Bus Stops." *Transportation Research Board 86th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 21–25.

Guenthner, R.P., Sinha, K.C., (1983). "Modeling bus delays due to passenger boardings and alightings." *Transportation Research Record*, No. 915, pp. 7-13.

Guenthner, R.P., Hamat, K., (1988). "Transit dwell time under complex fare structure." *Journal of Transportation Engineering*, Vol. 114, No. 3, pp. 367-379.

Hofmann, M., O'Mahony, M., (2005). "The Impact of Adverse Weather Conditions on Urban Bus Performance Measures." *Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems*, pp. 431-436. Vienna, Austria, September 13-16.

Hourdakis, J., Michalopoulos, P., Kottommannil, J., (2003). "Practical procedure for calibrating microscopic traffic simulation models." *Transportation Research Record*, No. 1852, pp. 130-139.

ITS America, (2002). *An Overview of Transit Signal Priority*. Advanced Traffic Management Systems Committee and Advanced Public Transportation Systems Committee of the ITS America, Washington, D.C.

ITS America, (2004). *An Overview of Transit Signal Priority – revised and updated*. Advanced Traffic Management Systems Committee and Advanced Public Transportation Systems Committee of the ITS America, Washington, D.C.

Jarmar Technologies, "JAMAR Hand-held traffic data collector, DB-400." Jarmar Technologies, Horsham, PA. http://www.jamartech.com/, accessed November 1, 2007.

Kim, W., Rilett, L.R., (2005). "An Improved Transit Signal Priority System for Networks With Nearside Bus Stops." *Transportation Research Board 84th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 9–13.

Kimpel, T.J., Strathman, J.G., Bertini, R.L., Callas, S., (2005). "Analysis of Transit Signal Priority Using Archived Tri-Met Bus Dispatch System Data." *Transportation Research Record*, No. 1925, pp. 156–166.

King County Department of Transportation, (2002). *An Evaluation of Transit Signal Priority in Aurora Avenue North*. Transit Speed and Reliability Program, King County Department of Transportation, King County, WA.

Kittelson & Associates, Inc., (2006). *LADOT/County Transit Signal Priority System*. Technical report. Kittelson & Associates, Inc., Portland, OR.

Levinson, H.S., (1983). "Analyzing transit travel time performance." *Transportation Research Record*, No. 915, pp. 1-6. http://pubsindex.trb.org/view.aspx?id=202380, accessed November 2010.

Li, M., Wu, G., Li, Y.I., Bu, F., Zhang, W.B., (2007). "Active Signal Priority for Light-Rail Transit at Grade Crossings." *Transportation Research Board 86th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 21–25.

Li, M., Yin, Y., Zhou, K., Zhang, W.B., Liu, H., Tan, C.W., (2005). "Adaptive Transit Signal Priority on Actuated Signalized Corridors." *Transportation Research Board 84th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 9–13.

Liao, C.F., Davis, G. (2007). "Simulation Study of Bus Signal Priority Strategy: Taking Advantage of Global Positioning System, Automated Vehicle Location System, and Wireless Communications." *Transportation Research Record*, No. 2034, pp. 82-91.

Liao, C.F., Davis, G.A., (2008). "An Adaptive Transit Signal Priority Strategy Using GPS/AVL and Wireless Communication Systems." *Proceedings of the 15th World Congress on Intelligent Transport Systems*, New York, NY, November 16-20.

Liao, C.F., Liu, H.X. (2010). "Development of a Data Processing Framework for Transit Performance Analysis." *Transportation Research Record*, No. 2143, pp. 34-43.

Lin, T.M., Wilson, N.H.M., (1992). "Dwell Time Relationships for Light Rail Systems." *Transportation Research Record*, No. 1361, pp. 287–295.

Liu, H., Skabardonis, A., Zhang, W.B., (2003). "A Dynamic Model for Adaptive Bus Signal Priority." *Transportation Research Board 82nd Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 12-16.

Liu, H., Skabardonis, A., Zhang, W.B., Li, M., (2004). "Optimal Detector Location for Bus Signal Priority." *Transportation Research Record*, No. 1867, pp. 144-150.

Liu, H., Wu, X., Ma, W., and Hu, H., (2009). "Real-Time queue length estimation for congested signalized intersections." *Transportation Research Part C*, Vol. 17, Issue 4, pp. 412-427.

Liu, H. and Ma, W., (2009). "A virtual vehicle probe model for time-dependent travel time estimation on signalized arterials." *Transportation Research Part C*, Vol. 17, Issue 1, pp. 11-26.

Marca, J.E., (2006). "Mobile throughput of 802.11b from a moving vehicle to a roadside access point." *Transportation Research Board 85th Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 22-26.

McLeod, F., Hounsell, N., (2003). "Bus Priority at Traffic Signals – Evaluating Strategy Options." *Journal of Public Transportation*, Vol. 6, No. 3, pp. 1-14. http://www.nctr.usf.edu/jpt/pdf/JPT%206-3.pdf, accessed October 1, 2007.

McNally, M.G., Marca, J.E., Rindt, C.R., Koos, A.M., (2003). *TRACER: In-vehicle, GPS-based, Wireless Technology for Traffic Surveillance and Management*. UCB-ITS-PRR-2003-23, California PATH, Berkeley, CA. http://www.path.berkeley.edu/PATH/Publications/PDF/PRR/2003/PRR-2003-23.pdf, accessed October 1, 2007.

Metro Transit, 2006. *Bottineau Corridor Transit Signal Priority: Conceptual Design*. Metro Transit, Metropolitan Council, Minneapolis, MN.

Milkovits, M.N., (2008). "Modeling the Factors Affecting Bus Stop Dwell Time: Use of Automatic Passenger Counting, Automatic Fare Counting, and Automatic Vehicle Location Data." *Transportation Research Record*, No. 2072, pp. 125-130.

Minnesota Department of Transportation, (2008). *Minnesota transit improvements through the Urban Partnership Agreement*. Minnesota Department of Transportation, St. Paul, MN. http://www.dot.state.mn.us/upa/transit.html, accessed July 2009.

Mirchandani, P.B., Head, K.L., (2001). "A real-time traffic signal control system: Architecture, algorithms and analysis." *Transportation Research Part C*, Vol. 9, No. 6, pp. 415-432.

Mirchandani, P.B., Knyazyan, A., Head, K.L., Wu, W., (2001). "An Approach towards the Integration of Bus Priority, Traffic Adaptive Signal Control, and Bus Information/Scheduling Systems." *Journal of Scheduling*, Vol. 505, pp. 319-334.

Mirchandani, P.B., Lucas, D.E., (2004). "Integrated Transit Priority and Rail/Emergency Preemption in Real-Time Traffic Adaptive Signal Control." *Journal of Intelligent Transportation Systems Technology, Planning and Operations*, Vol. 8, Issue 2, pp. 101-115. http://www.informaworld.com/smpp/content~content=a713904020~db=all~order=page, accessed October 1, 2007.

Puong, A., (2000). "Dwell Time Model and Analysis for the MBTA Red Line." Internal memo, MIT, http://dspace.mit.edu/bitstream/handle/1721.1/35716/1-258JFall-2003/NR/rdonlyres/Civil-and-Environmental-Engineering/1-258JPublic-Transportation-Service-and-Operations-PlanningFall2003/D9613FBC-9279-4F31-A46D-8DB2E037E9E4/0/a3_dwelltim.pdf, accessed November 2010.

Rakha, H., Ahn K., Collura J., (2006). *Transit Signal Priority Project Along Route 1: Lessons Learned*. VTRC 06-CR, Virginia Transportation Research Council, Charlottesville, VA.

Saint Cloud Metropolitan Transit Commission, (2000). *Transit Priority Evaluation Report, Final Report*. Metropolitan Transit Commission, St. Cloud, MN.

Siemens Traffic Controls, (2006). *SCOOT, Split Cycle Offset Optimisation Technique*. www.scoot-utc.com, accessed October 1, 2007.

Siemens Intelligent Transportation Systems, (2002). *SEPAC Actuated Signal Control Software, User's manual*. http://www.itssiemens.com/en/t_nav228.html, accessed October 1, 2007.

Siemens Intelligent Transportation Systems, (2006). *Eagle EPAC M50 Traffic Control Unit*, http://www.siemens.com.co/SiemensDotNetClient_Andina/Medias/PDFS/395_20080303193238.pdf, accessed October 17, 2011.

Skabardonis, A., (2000). "Control Strategies for Signal Priority." *Transportation Research Record*, No. 1727, pp. 20-26.

Skabardonis, A., Geroliminis, N., (2008). "Real-Time Monitoring and Control on Signalized Arterials." *Journal of Intelligent Transportation Systems*, Vol. 12, Issue 2, pp. 64–74.

Stibor, L., Zang, Y., Reumerman, H., (2007). "Neighborhood evaluation of vehicular ad-hoc network using IEEE 802.11p." *Compendium of 13th European Wireless Conference*, Paris, France, April 1-4. http://www.ew2007.org/papers/1569014956.pdf, accessed October 1, 2007.

Torrent-Moreno, M., Jiang, D., Hartenstein, H., (2004). "Broadcast reception rates and effects of priority access in 802.11-based vehicular ad-hoc networks". *Proceedings of the 1st ACM International Workshop on Vehicular Ad Hoc Networks*, Association for Computing Machinery, New York, NY, pp. 10-18.

Trafficware Corporation, Synchro, (2006). *Traffic signal coordination software*. Trafficware Corporation, Albany, CA. http://www.trafficware.com/, accessed October 1, 2007.

Transportation Research Board, (2003). "Part 4: Bus Transit Capacity, Chapter 1, Bus Capacity Fundamentals." *Transit Cooperative Research Program (TCRP) Report 100: Transit Capacity and Quality Service Manual*, 2nd Edition. pp. 4-3 - 4-9. http://onlinepubs.trb.org/onlinepubs/tcrp/tcrp100/part%200.pdf, accessed October 1, 2008.

Transport Simulation Systems, (2002). *AIMSUN Version 4.1 User's Manual*. Transport Simulation Systems (TSS), Barcelona, Spain. http://www.aimsun.com/site/, accessed October 1, 2007.

Transport Simulation Systems, (2002). *GETRAM Extensions Version 4.1 User's Manual*. TSS, Barcelona, Spain.

Wadjas, Y., Furth, P.G., (2003). "Transit Signal Priority along Arterials Using Advanced Detection." *Transportation Research Record*, No. 1856, pp. 220-230.

Wischhof, L., Ebner, A., Rohling, H., Lott, M., Hafmann, R., (2003). "Adaptive Broadcast for Travel and Traffic Information Distribution Based on Inter-Vehicle Communication." *Proceedings of IEEE Vehicular Technology Conference (VTC) 2003*, Orlando, FL, October 6-9.

Wu, H., Lee, J., Hunter, M., Fujimoto, R., Guensler, R., Ko, J., (2005). "Efficiency of Simulating Vehicle-Vehicle Message Propagation in Atlanta, Georgia, I-75 Corridor." *Transportation Research Record*, No. 1910, pp. 82-89.

Xu, Q., Hedrick, K., Sengupta, R., VanderWerf, J., (2002). "Effects of Vehicle-vehicle/ roadside-vehicle Communication on Adaptive Cruise Controlled Highway Systems." *Proceedings of IEEE 56th Vehicular Technology Conference*, Vol. 2, pp. 1249- 1253, Birmingham, AL, May.

Zhang, J., (2003). "Zero Public Infrastructure Vehicle-Based Traffic Information System." *Transportation Research Board 82nd Annual Meeting Compendium of Papers*, CD-ROM, Washington, D.C., January 12-16.

Zheng, J., Wang, Y., Liu, H., Hallenbeck, M.E., (2007). "Modeling Impact of Nearside Bus Stop on Transit Delays at Transit Signal Priority Enabled Intersections." *Transportation Research Board 86th Annual Meeting Compendium of Papers*, CD-ROM, Washington D.C., January 21-25.

APPENDIX A: SIGNAL PHASING AND TIMING INFORMATION

A.1 Route 10 Intersections

Here is the list of intersections on Central Avenue (TH65) and University Avenue (TH 47) that bus RTE 10 travels through.

Table A.1. List of TH65 Intersections

Location ID	Located On	Between/At	Agency
388	3rd Av Bridge	1st St S AND 2nd St SE	MPLS
389	Central Av SE	4th St SE AND 5th St SE	MPLS
258	Central Av NE	3rd Av NE AND Spring St NE	MPLS
259	Central Av NE	Spring St NE AND Broadway St NE	MPLS
260	Central Av NE	Broadway St NE AND 12th Av NE	MPLS
5261	Central Av NE	Lowry Av NE AND 26th Av NE	MPLS
9262	Central Av NE	Columbia Pkwy AND 37th Av NE	MPLS
3E-02-9866	Central Av NE	S OF 49TH AVE NE / CSAH-4	MNDOT
3E-02-10399	Central Av NE	N OF 37th AV NE IN COLUMBIA HTS	MNDOT
3E-02-10401	Central Av NE	S OF 44TH AVE NE.	MNDOT
3F-02-11221	Central Av NE	S OF 53RD AVE NE	MNDOT
3F-02-10405	TH 65	N OF I-694 IN FRIDLEY	MNDOT
3F-02-10406	TH 65	S OF MISSISSIPPI AVE./CSAH-6	MNDOT
3F-02-10408	TH 65	N OF 68TH AVE.	MNDOT
3F-02-11435	TH-65	S OF I-694	MNDOT
3F-02-10403	TH 47	S OF I-694 IN FRIDLEY	MNDOT
3F-02-10404	TH 47	N OF I-694 IN FRIDLEY	MNDOT
3F-02-10409	TH-47	S OF 73RD AVE NE.	MNDOT
3F-02-10466	TH-47	S OF 53RD AVE NE	MNDOT
3F-02-10414	TH-47	S OF CO.RD.3 / 85TH AVE.	MNDOT
3F-02-10407	TH-47	S OF MISSISSIPPI ST.	MNDOT

A.2 Minneapolis Intersections

Table A.2. Off-Peak Timing and Phasing Assignment – Minneapolis Intersections

Intersection #	Controller	Intersection	OFF-PEAK											
			Cycle	Split (%)									Offset	
			(sec)	#	1	2	3	4	5	6	7	8	(%)	
858	EF-20	Central Ave SE & 2nd St	70	4	65	35	0	0	0	0	0	0	4	58
397	EF-120	Central Ave & University Ave SE	70	4	47	35	18	0	0	0	0	0	4	67
4	EF-20	Central Ave SE & 4th St	70	4	52	48	0	0	0	0	0	0	4	49
883	HMC-1000	Central Ave, E Hennepin Ave & 5th St	70	4	0	38	29	33	0	38	0	62	4	39
328	EF-20	Central Ave, 1st Ave NE & 7th St	70	4	67	33	0	0	0	0	0	0	4	34
129	EPAC, act	Central Ave NE & Spring St	90	4	36	48	16	0	0	0	0	0	4	39
498	EPAC, act	Broadway St & Central Ave NE	90	4	17	33	17	33	17	33	17	33	4	53
335	EPAC, act	Central Ave & 14th Ave NE	70	4	57	43	0	0	0	0	0	0	4	10
446	EPAC, pre	Central Ave & 18th Ave NE	70	4	40	60	0	0	0	0	0	0	4	61
229	EPAC, pre	Central Ave & 18-1/2 Ave NE	70	4	52	48	0	0	0	0	0	0	4	55
230	EF-20	Central Ave & 19th Ave NE	70	4	61	39	0	0	0	0	0	0	4	1
231	EF-20	Central Ave & 20th Ave NE	70	4	41	59	0	0	0	0	0	0	4	9
232	EF-20	Central Ave & 22nd Ave NE	70	4	54	46	0	0	0	0	0	0	4	13
448	EF-20	Central Ave & 24th Ave NE	70	4	65	35	0	0	0	0	0	0	4	49
398	EF-20	Central Ave & Lowry Ave NE	70	4	41	16	43	0	0	0	0	0	4	6
449	EF-20	Central Ave & 26th Ave NE	70	4	61	39	0	0	0	0	0	0	4	54
59	EF-20	Central Ave & 27th Ave NE	70	4	52	48	0	0	0	0	0	0	4	15
884	EPAC	Central Ave & 28th Ave NE	70	4	60	40	0	0	0	0	0	0	4	4
801	EF-20	Central Ave NE & St Anthony Pkwy	70	4	65	35	0	0	0	0	0	0	4	4
400	HMP-40	Central Ave & 34th Ave NE	70	4	58	42	0	0	0	0	0	0	4	17
948	EPAC, act	Central Ave, Reservoir Blvd & 37th Ave NE	free	0	0	0	0	0	0	0	0	0	0	0

Table A.3. AM-Peak Timing and Phasing Assignment – Minneapolis Intersections

Intersection #	Controller	Intersection	AM-PEAK (06:00-08:45)											
			Cycle	Split (%)									Offset	
			(sec)	#	1	2	3	4	5	6	7	8	(%)	
858	EF-20	Central Ave SE & 2nd St	80	8	70	30	0	0	0	0	0	0	8	53
397	EF-120	Central Ave & University Ave SE	80	8	35	45	20	0	0	0	0	0	8	65
4	EF-20	Central Ave SE & 4th St	80	8	59	41	0	0	0	0	0	0	8	41
883	HMC-1000	Central Ave, E Hennepin Ave & 5th St	80	8	0	46	25	29	0	46	0	54	8	25
328	EF-20	Central Ave, 1st Ave NE & 7th St	80	8	65	35	0	0	0	0	0	0	8	20
129	EPAC, act	Central Ave NE & Spring St	0	8	36	48	16	0	0	0	0	0	8	84
498	EPAC, act	Broadway St & Central Ave NE	0	8	17	33	17	33	17	33	17	33	8	33
335	EPAC, act	Central Ave & 14th Ave NE	80	8	62	38	0	0	0	0	0	0	8	85
446	EPAC, pre	Central Ave & 18th Ave NE	80	8	55	45	0	0	0	0	0	0	8	69
229	EPAC, pre	Central Ave & 18-1/2 Ave NE	80	8	58	42	0	0	0	0	0	0	8	61
230	EF-20	Central Ave & 19th Ave NE	80	8	65	35	0	0	0	0	0	0	8	54
231	EF-20	Central Ave & 20th Ave NE	80	8	70	30	0	0	0	0	0	0	8	45
232	EF-20	Central Ave & 22nd Ave NE	80	8	60	40	0	0	0	0	0	0	8	24
448	EF-20	Central Ave & 24th Ave NE	80	8	70	30	0	0	0	0	0	0	8	11
398	EF-20	Central Ave & Lowry Ave NE	80	8	31	14	55	0	0	0	0	0	8	66
449	EF-20	Central Ave & 26th Ave NE	80	8	66	34	0	0	0	0	0	0	8	96
59	EF-20	Central Ave & 27th Ave NE	80	8	58	42	0	0	0	0	0	0	8	78
884	EPAC	Central Ave & 28th Ave NE	80	8	65	35	0	0	0	0	0	0	8	71
801	EF-20	Central Ave NE & St Anthony Pkwy	80	8	65	35	0	0	0	0	0	0	8	3
400	HMP-40	Central Ave & 34th Ave NE	80	8	63	37	0	0	0	0	0	0	8	88
948	EPAC, act	Central Ave, Reservoir Blvd & 37th Ave NE	free	0	0	0	0	0	0	0	0	0	0	0

Table A.4 PM-Peak Timing and Phasing Assignment – Minneapolis Intersections

Intersection #	Controller	Intersection	PM-PEAK (15:00-18:30)											
			Cycle	Split (%)									Offset	
			(sec)	#	1	2	3	4	5	6	7	8	(%)	
858	EF-20	Central Ave SE & 2nd St	80	9	70	30	0	0	0	0	0	0	9	34
397	EF-120	Central Ave & University Ave SE	80	9	48	34	18	0	0	0	0	0	9	63
4	EF-20	Central Ave SE & 4th St	80	9	52	48	0	0	0	0	0	0	9	53
883	HMC-1000	Central Ave, E Hennepin Ave & 5th St	80	9	23	35	0	42	0	58	0	42	9	55
328	EF-20	Central Ave, 1st Ave NE & 7th St	80	9	65	35	0	0	0	0	0	0	9	75
129	EPAC, act	Central Ave NE & Spring St	0	9	36	48	16	0	0	0	0	0	9	31
498	EPAC, act	Broadway St & Central Ave NE	0	9	17	33	17	33	17	33	17	33	9	60
335	EPAC, act	Central Ave & 14th Ave NE	80	9	62	38	0	0	0	0	0	0	9	61
446	EPAC, pre	Central Ave & 18th Ave NE	80	9	55	45	0	0	0	0	0	0	9	11
229	EPAC, pre	Central Ave & 18-1/2 Ave NE	80	9	52	48	0	0	0	0	0	0	9	3
230	EF-20	Central Ave & 19th Ave NE	80	9	70	30	0	0	0	0	0	0	9	30
231	EF-20	Central Ave & 20th Ave NE	80	9	70	30	0	0	0	0	0	0	9	41
232	EF-20	Central Ave & 22nd Ave NE	80	9	60	40	0	0	0	0	0	0	9	33
448	EF-20	Central Ave & 24th Ave NE	80	9	70	30	0	0	0	0	0	0	9	70
398	EF-20	Central Ave & Lowry Ave NE	80	9	44	14	42	0	0	0	0	0	9	30
449	EF-20	Central Ave & 26th Ave NE	80	9	66	34	0	0	0	0	0	0	9	81
59	EF-20	Central Ave & 27th Ave NE	80	9	58	42	0	0	0	0	0	0	9	25
884	EPAC	Central Ave & 28th Ave NE	80	9	65	35	0	0	0	0	0	0	9	39
801	EF-20	Central Ave NE & St Anthony Pkwy	80	9	67	33	0	0	0	0	0	0	9	79
400	HMP-40	Central Ave & 34th Ave NE	80	9	63	37	0	0	0	0	0	0	9	28
948	EPAC, act	Central Ave, Reservoir Blvd & 37th Ave NE	free	0	0	0	0	0	0	0	0	0	0	0

A.3 MnDOT Intersections

All MnDOT intersections along Central Avenue are actuated signal controllers.

Table A.5 AM & PM Peak Green Time (sec) – MnDOT Intersections

S_ID	S_ADDR	GRN_1	GRN_2	GRN_3	GRN_4	GRN_5	GRN_6	GRN_7	GRN_8
21147	HWY 65 & (2) 40TH AV NE	7	15	5	10	7	15	5	10
21148	HWY 65 & 41ST AV NE	7	15		10	7	15		10
21149	HWY 65 & 44TH AV NE	5	15		8	5	15		8
21150	HWY 65 & 45TH AV NE	7	15		10	7	15		10
21151	HWY 65 & 47TH AV NE		15		10	7	15		
21153	HWY 65 & (4) 49TH AV NE	7	15	5	10	7	15	5	10
21154	HWY 65 & 50TH AV NE	7	15		10	7	15		10
21155	HWY 65 & 52ND AV NE	7	15		10	7	15		10
21156	HWY 65 & 53RD AVE NE	7	15	8	8	7	15		

Table A.6 AM & PM Peak Yellow Time (sec) – MnDOT Intersections

S_ID	S_ADDR	YLW_1	YLW_2	YLW_3	YLW_4	YLW_5	YLW_6	YLW_7	YLW_8
21147	HWY 65 & (2) 40TH AV NE	3	3.5	3	3.5	3	3.5	3	3.5
21148	HWY 65 & 41ST AV NE	3	3.5		3.5	3	3.5		3.5
21149	HWY 65 & 44TH AV NE	3	3.5		3.5	3	3.5		3.5
21150	HWY 65 & 45TH AV NE	3	4		3.5	3	4		3.5
21151	HWY 65 & 47TH AV NE		4		3.5	3	4		
21153	HWY 65 & (4) 49TH AV NE	3	4	3	3.5	3	4	3	3.5
21154	HWY 65 & 50TH AV NE	3	4		3.5	3	4		3.5
21155	HWY 65 & 52ND AV NE	4	4		3.5	4	4		3.5
21156	HWY 65 & 53RD AVE NE	3	4	3.5	3.5	3	4		

Table A.7 AM & PM Peak Red Time (sec) – MnDOT Intersections

S_ID	S_ADDR	RED_1	RED_2	RED_3	RED_4	RED_5	RED_6	RED_7	RED_8
21147	HWY 65 & (2) 40TH AV NE	2	1.5	2	2	2	1.5	2	2
21148	HWY 65 & 41ST AV NE	2	1.5		2	2	1.5		2
21149	HWY 65 & 44TH AV NE	2	1.5		2	2	1.5		2
21150	HWY 65 & 45TH AV NE	2	1.5		2	2	1.5		2
21151	HWY 65 & 47TH AV NE		1.5		2.5	2	1.5		
21153	HWY 65 & (4) 49TH AV NE	2	1.5	2	2.5	2	1.5	2	2.5
21154	HWY 65 & 50TH AV NE	2	1.5		2.5	2	1.5		2.5
21155	HWY 65 & 52ND AV NE	2	1.5		2.5	2	1.5		2.5
21156	HWY 65 & 53RD AVE NE	2	2	2.5	2.5	2	2		

APPENDIX B: ROUTE 10 LINK VOLUME

Table B.1. Node CEUN Link Volume by Hour

Node	hr	actuate	dir	cycle	split	lanes	link_vol
CEUN	5	0	N	70	38.5	2	70
CEUN	6	0	N	80	29.6	2	150.5
CEUN	7	0	N	80	29.6	2	403
CEUN	8	0	N	80	29.6	2	569.1667
CEUN	9	0	N	70	38.5	2	444
CEUN	10	0	N	70	38.5	2	438
CEUN	11	0	N	70	38.5	2	484.5
CEUN	12	0	N	70	38.5	2	603.3333
CEUN	13	0	N	70	38.5	2	546.1667
CEUN	14	0	N	70	38.5	2	611
CEUN	15	0	N	80	48.8	2	720.6667
CEUN	16	0	N	80	48.8	2	916.1667
CEUN	17	0	N	80	48.8	2	969.8333
CEUN	18	0	N	80	48.8	2	636.5
CEUN	19	0	N	70	38.5	2	448.1667
CEUN	20	0	N	70	38.5	2	333.5
CEUN	21	0	N	70	38.5	2	314.8333
CEUN	5	0	S	70	38.5	2	133.6667
CEUN	6	0	S	80	29.6	2	415.5
CEUN	7	0	S	80	29.6	2	754.1667
CEUN	8	0	S	80	29.6	2	668.6667
CEUN	9	0	S	70	38.5	2	485
CEUN	10	0	S	70	38.5	2	411.8333
CEUN	11	0	S	70	38.5	2	465.6667
CEUN	12	0	S	70	38.5	2	550.3333
CEUN	13	0	S	70	38.5	2	528.5
CEUN	14	0	S	70	38.5	2	525.1667
CEUN	15	0	S	80	48.8	2	529.3333
CEUN	16	0	S	80	48.8	2	553.6667
CEUN	17	0	S	80	48.8	2	569.3333
CEUN	18	0	S	80	48.8	2	503.5
CEUN	19	0	S	70	38.5	2	389.3333
CEUN	20	0	S	70	38.5	2	330.6667
CEUN	21	0	S	70	38.5	2	287

Table B.2. Node LWCE Link Volume by Hour

Node	hr	actuate	dir	cycle	split	lanes	link_vol
LWCE	5	0	N	70	30.1	2	46.75
LWCE	6	0	N	80	30.4	2	97
LWCE	7	0	N	80	30.4	2	178.5
LWCE	8	0	N	80	30.4	2	231.75
LWCE	9	0	N	70	30.1	2	282.75
LWCE	10	0	N	70	30.1	2	342.5
LWCE	11	0	N	70	30.1	2	425
LWCE	12	0	N	70	30.1	2	453.75
LWCE	13	0	N	70	30.1	2	467.5
LWCE	14	0	N	70	30.1	2	499.25
LWCE	15	0	N	80	32.8	2	633.5
LWCE	16	0	N	80	32.8	2	711.25
LWCE	17	0	N	80	32.8	2	720.75
LWCE	18	0	N	80	32.8	2	497.25
LWCE	19	0	N	70	30.1	2	348.75
LWCE	20	0	N	70	30.1	2	287.5
LWCE	21	0	N	70	30.1	2	234.75
LWCE	5	0	S	70	30.1	2	126.25
LWCE	6	0	S	80	30.4	2	358.75
LWCE	7	0	S	80	30.4	2	547.25
LWCE	8	0	S	80	30.4	2	465.25
LWCE	9	0	S	70	30.1	2	342.5
LWCE	10	0	S	70	30.1	2	354
LWCE	11	0	S	70	30.1	2	371.25
LWCE	12	0	S	70	30.1	2	439.5
LWCE	13	0	S	70	30.1	2	433.5
LWCE	14	0	S	70	30.1	2	432.25
LWCE	15	0	S	80	32.8	2	405.75
LWCE	16	0	S	80	32.8	2	430
LWCE	17	0	S	80	32.8	2	403.25
LWCE	18	0	S	80	32.8	2	412.5
LWCE	19	0	S	70	30.1	2	318
LWCE	20	0	S	70	30.1	2	271.5
LWCE	21	0	S	70	30.1	2	225.75

Table B.3. Node 41CE Link Volume by Hour

Node	hr	actuate	dir	cycle	split	lanes	link_vol
41CE	5	1	N	100	25	2	168
41CE	6	1	N	100	25	2	290
41CE	7	1	N	120	30	2	475
41CE	8	1	N	120	30	2	473
41CE	9	1	N	120	30	2	512
41CE	10	1	N	100	25	2	638
41CE	11	1	N	100	25	2	730
41CE	12	1	N	135	33.75	2	827
41CE	13	1	N	135	33.75	2	757
41CE	14	1	N	135	33.75	2	862
41CE	15	1	N	135	33.75	2	987
41CE	16	1	N	135	33.75	2	1143
41CE	17	1	N	135	33.75	2	1050
41CE	18	1	N	135	33.75	2	785
41CE	19	1	N	135	33.75	2	573
41CE	20	1	N	100	25	2	590
41CE	21	1	N	100	25	2	333
41CE	5	1	S	100	25	2	232
41CE	6	1	S	100	25	2	640
41CE	7	1	S	120	30	2	957
41CE	8	1	S	120	30	2	809
41CE	9	1	S	120	30	2	681
41CE	10	1	S	100	25	2	695
41CE	11	1	S	100	25	2	725
41CE	12	1	S	135	33.75	2	758
41CE	13	1	S	135	33.75	2	743
41CE	14	1	S	135	33.75	2	740
41CE	15	1	S	135	33.75	2	771
41CE	16	1	S	135	33.75	2	774
41CE	17	1	S	135	33.75	2	718
41CE	18	1	S	135	33.75	2	701
41CE	19	1	S	135	33.75	2	550
41CE	20	1	S	100	25	2	468
41CE	21	1	S	100	25	2	340

Table B.4. Node 51CE Link Volume by Hour

Node	hr	actuate	dir	cycle	split	lanes	link_vol
51CE	5	1	N	100	25	2	313
51CE	6	1	N	100	25	2	522
51CE	7	1	N	120	30	2	637
51CE	8	1	N	120	30	2	625
51CE	9	1	N	120	30	2	661
51CE	10	1	N	100	25	2	772
51CE	11	1	N	100	25	2	826
51CE	12	1	N	135	33.75	2	955
51CE	13	1	N	135	33.75	2	925
51CE	14	1	N	135	33.75	2	976
51CE	15	1	N	160	40	2	1105
51CE	16	1	N	160	40	2	1159
51CE	17	1	N	160	40	2	1205
51CE	18	1	N	135	33.75	2	1022
51CE	19	1	N	135	33.75	2	762
51CE	20	1	N	100	25	2	717
51CE	21	1	N	100	25	2	474
51CE	5	1	S	100	25	2	263
51CE	6	1	S	100	25	2	722
51CE	7	1	S	120	30	2	1080
51CE	8	1	S	120	30	2	871
51CE	9	1	S	120	30	2	808
51CE	10	1	S	100	25	2	783
51CE	11	1	S	100	25	2	969
51CE	12	1	S	135	33.75	2	1014
51CE	13	1	S	135	33.75	2	925
51CE	14	1	S	135	33.75	2	954
51CE	15	1	S	160	40	2	1016
51CE	16	1	S	160	40	2	1051
51CE	17	1	S	160	40	2	1027
51CE	18	1	S	135	33.75	2	1006
51CE	19	1	S	135	33.75	2	780
51CE	20	1	S	100	25	2	710
51CE	21	1	S	100	25	2	514

APPENDIX C: TS-7800 EMBEDDED COMPUTER SYSTEM

C.1 TS-7800 Documentation Online Links

- TS-7800 Datasheet
- TS-7800 Schematic
- TS-7800 Mechanical Drawing
- Linux for ARM on TS-72XX User's Guide
- TS-7800 Manual
- TS-7800 Kernel Compile Guide

C.2 TS-7800 Recovery

Reimage SD Card
Although it is easy to get your board into an unbootable state during development if you botch a modification, it is equally easy to use an SD card to completely restore the onboard flash. First, a properly formatted SD card must be created, the latest image can be downloaded from ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7800-linux/binaries/ts-images/512mbsd-latest.dd.bz2
Please note that this image does not contain the Eclipse IDE that is distributed with 2GB cards from Technologic Systems. Once the image has been downloaded it can be copied to the SD card with the following command (assuming /dev/sda is the device where the SD card is located).

```
bzcat 512mbsd-latest.dd.bz2 | dd of=/dev/sda
```

or

```
bunzip2 512mbsd-latest.dd.bz2
dd if=512mbsd-latest.dd of=/dev/sda
```

Place a jumper on JP1 to make the TS-7800 boot from SD. Within a few seconds the board will have booted to a fastboot prompt from the SD card.

C.2.1 Reimaging the onboard flash from the SD card

The commands 'createmtdroot' and 'createmtdboot' can then be used to restore the onboard NAND flash to its factory settings.

C.3 TS-7800 Auto Run

Run startup of the TS-7800 module when power up:
The ts-7800 module uses a script called “linuxrc” which is located in the root (/) directory. To put the whole application in startup, we need to include the 2 commands of compiling and running in the linuxrc file. To do this:

1) Make sure you have all the codes and the compiled output (e.g., a.out file) in SD card.
2) Remove the SD card from the slot and restart the TS-7800 module.
3) Now set the linuxrc file to linuxrc-mtdroot by the following command:
    a. ln -sf linuxrc-mtdroot linuxrc
    b. save
4) This sets the boot mode to onboard flash.
5) Now put the SD card without the jumper and restart the module. This boots you into the onboard flash.
6) Now mount the SD card, by the following command:
    a. mount /dev/tssdcard3 /mnt/host
7) Once the SD card is mounted, open the /linuxrc file which is on the SD card as follows:
    a. cd /mnt/host
    b. vi linuxrc-sdroot
8) There is a line which says: “hack”. Before this line, and after the line “`export ENV=/shinit`”, insert the line
    a. /mnt/root/home/varun/a.out > $CONSOLE
9) The “$CONSOLE” part of it, directs all the output messages (like errors/warnings and the output of the program) to be displayed on screen.
10) Save and close the file.
11) Then do a sync on the system as follows:
    a. sync
12) Now link the linuxrc to the linuxrc-sdroot file as follows:
    a. ln -sf linuxrc-sdroot linuxrc
    b. save
13) Now restart the module with the jumper-JP1 on and the SD card in its slot. Check whether the code runs in the startup or not.

NOTE: Removing the jumper JP1 and keeping the SD card in the slot, makes the module boot from the onboard flash.

Instructions to recover the log data files from the SD card

1) Remove the jumper-JP1 and restart the ts-7800 module, with the SD card in the slot.
2) The SD card needs to be mounted by the following command:
    a. mount /dev/tssdcard3 /mnt/host
3) Now the linuxrc file has to be changed to comment out the program execution line from the startup. This can be done as follows:
    a. Open the file `/mnt/host/linuxrc-sdroot`.
    b. Comment out the line “/mnt/root/home/varun/a.out>$CONSOLE”
    c. Save the file. Sync the system.
4) Now restart the module with jumper- JP1 on.
5) Now you will find the files in /mnt/root/home/varun folder.

APPENDIX D: BUS ROUTE 10 DATA

D.1 Route #10 Map

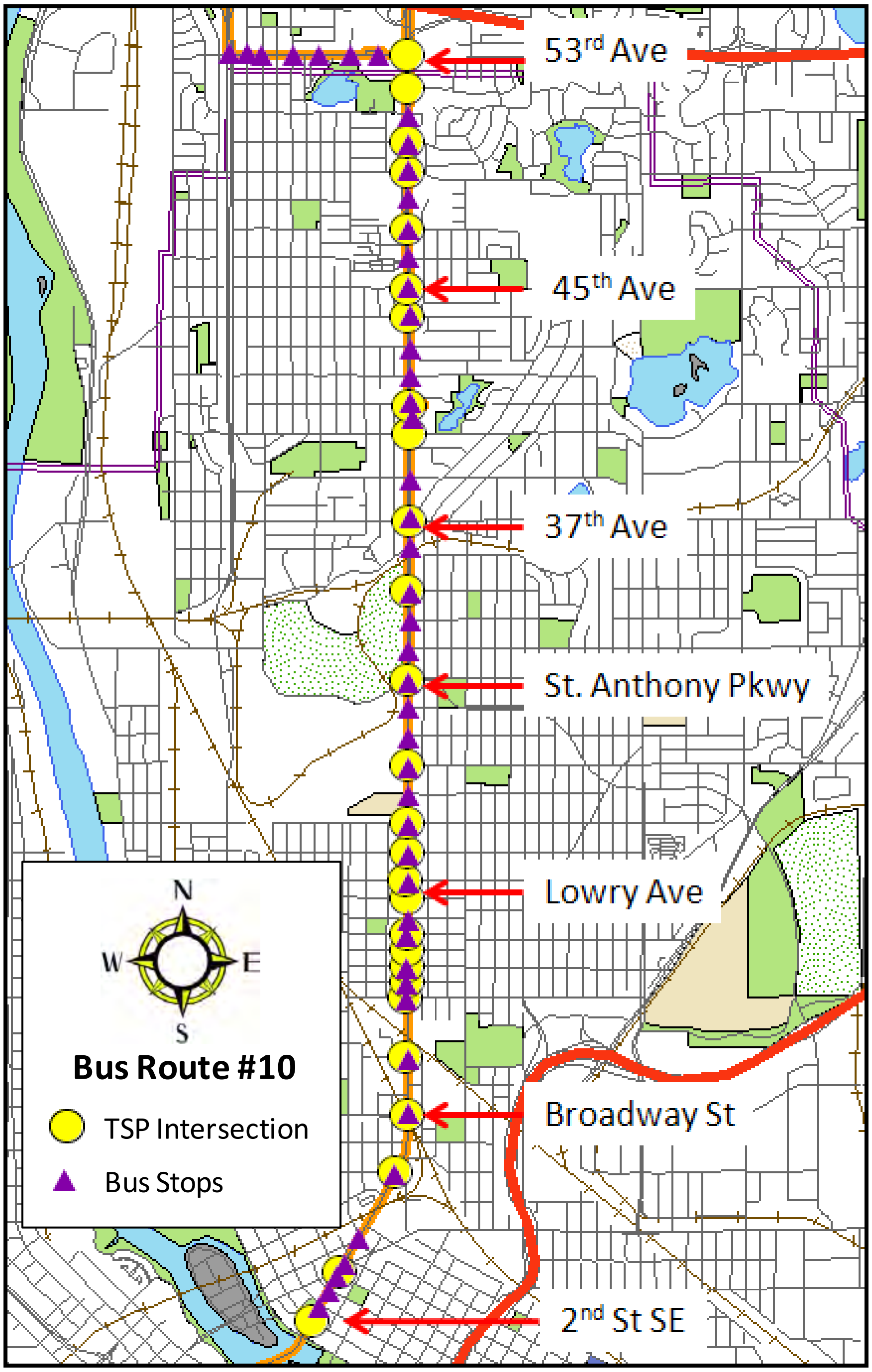

Figure D.1. Map of Route #10

D.2 Intersections of Bus Route #10

Table D.1. Route #10 TSP Intersections

ID	Intersection On	Intersection At	TSP_Equipped
1	Central Ave	53th Ave NE	1
2	Central Ave	52th Ave NE	1
3	Central Ave	50th Ave NE	1
4	Central Ave	49th Ave NE	1
5	Central Ave	47th Ave NE	1
6	Central Ave	45th Ave NE	1
7	Central Ave	44th Ave NE	1
8	Central Ave	41st Ave NE	1
9	Central Ave	40th Ave NE	1
10	Central Ave	37th Ave NE	1
11	Central Ave	35th Ave NE	1
12	Central Ave	St Anthony Pkwy	1
13	Central Ave	29th Ave NE	1
14	Central Ave	27th Ave NE	1
15	Central Ave	26th Ave NE	1
16	Central Ave	Lowry	1
17	Central Ave	24th Ave NE	1
18	Central Ave	22nd Ave NE	1
19	Central Ave	20th Ave NE	1
20	Central Ave	19th Ave NE	1
21	Central Ave	18th half Ave NE	1
22	Central Ave	18th Ave NE	1
23	Central Ave	14th Ave NE	1
24	Central Ave	Broadway St NE	1
25	Central Ave	Spring St NE	1
26	Central Ave	Hennepin and 5th	1
27	Central Ave	2nd St SE	1

D.3 Stops of Bus Route #10

Table D.2. List of route #10 NB stops

Stop_seq	Site_ID	Site_on	Site_at	Node_ID	Corner_desc
1	42837	LEAMINGTON RAMP	UPPER LEVEL	LMRP	NEAR SIDE
2	52104	11 ST S	MARQUETTE AV S	1 2A	FAR SIDE
3	17991	NICOLLET MALL	10 ST S		NEAR SIDE
4	17992	NICOLLET MALL	9 ST S		NEAR SIDE
5	17993	NICOLLET MALL	8 ST S	7S2A	NEAR SIDE
6	17994	NICOLLET MALL	7 ST S	7SNI	NEAR SIDE
7	17995	NICOLLET MALL	6 ST S		NEAR SIDE
8	17996	NICOLLET MALL	6 ST / 5 ST S		MID BLOCK
9	17997	NICOLLET MALL	4 ST S	4NIC	NEAR SIDE
10	17998	NICOLLET MALL	4 ST / 3 ST S	3SNI	MID BLOCK
11	17999	NICOLLET MALL	WASHINGTON AV S		NEAR SIDE
12	19315	WASHINGTON AV S	MARQUETTE AV S	MA2S	NEAR SIDE
13	51382	3 AV S	WASHINGTON AV		FAR SIDE
14	14952	CENTRAL AV	UNIVERSITY AV SE	CEUN	NEAR SIDE
15	14953	CENTRAL AV	4 ST SE		NEAR SIDE
16	14954	CENTRAL AV	5 ST SE		NEAR SIDE
17	40953	CENTRAL AV NE	HENNEPIN E / 6 ST		MID BLOCK
18	17224	CENTRAL AV NE	7 ST SE		FAR SIDE
19	17219	CENTRAL AV NE	SPRING ST		NEAR SIDE
20	17215	CENTRAL AV NE	BROADWAY ST		FAR SIDE
21	17211	CENTRAL AV NE	14 AV NE		NEAR SIDE
22	17209	CENTRAL AV NE	18 AV NE		NEAR SIDE
23	17206	CENTRAL AV NE	18 1/2 AV NE		NEAR SIDE
24	17205	CENTRAL AV NE	19 AV NE		NEAR SIDE
25	17201	CENTRAL AV NE	22 AV NE		NEAR SIDE
26	17198	CENTRAL AV NE	23 AV NE		NEAR SIDE
27	17194	CENTRAL AV NE	LOWRY AV	LWCE	NEAR SIDE
28	52073	CENTRAL AV NE	26 AV NE		NEAR SIDE
29	17190	CENTRAL AV NE	27 AV NE		NEAR SIDE
30	17189	CENTRAL AV NE	28 AV NE		NEAR SIDE
31	17186	CENTRAL AV NE	29 AV NE		NEAR SIDE
32	17185	CENTRAL AV NE	30 AV NE		NEAR SIDE
33	17182	CENTRAL AV NE	31 AV NE		NEAR SIDE
34	17181	CENTRAL AV NE	ST ANTHONY PKWY		NEAR SIDE
35	17178	CENTRAL AV NE	33 AV NE		NEAR SIDE
36	17177	CENTRAL AV NE	34 AV NE		NEAR SIDE

37	17174	CENTRAL AV NE	35 AV NE		NEAR SIDE
38	17172	CENTRAL AV NE	COLUMBIA PKWY		ACROSS FROM
39	17169	CENTRAL AV NE	37 AV NE		FAR SIDE
40	17168	CENTRAL AV NE	39 AV NE		NEAR SIDE
41	17164	COL HTS TRANSIT CTR	40 AV / 41 AV NE	4 CE	MID BLOCK
42	17163	CENTRAL AV NE	41 AV NE	41CE	FAR SIDE
43	17160	CENTRAL AV NE	42 AV		NEAR SIDE
44	17159	CENTRAL AV NE	43 AV		NEAR SIDE
45	17156	CENTRAL AV NE	44 AV NE		FAR SIDE
46	17155	CENTRAL AV NE	45 AV		FAR SIDE
47	17152	CENTRAL AV NE	46 AV		FAR SIDE
48	17151	CENTRAL AV NE	47 AV		FAR SIDE
49	17148	CENTRAL AV NE	#4801		FAR SIDE
50	17147	CENTRAL AV NE	49 AV		FAR SIDE
51	17144	CENTRAL AV NE	50 AV		NEAR SIDE
52	17143	CENTRAL AV NE	51 COURT	51CE	NEAR SIDE
53	14163	53 AV NE	TARGET - E ENTRANCE	53TA	FAR SIDE
54	14165	53 AV NE	MONROE ST NE		NEAR SIDE
55	14170	53 AV NE	SULLIVAN DR		ACROSS FROM
56	14174	53 AV NE	7 ST		NEAR SIDE
57	14178	53 AV NE	5 ST		NEAR SIDE
58	14179	53 AV NE	4 ST		NEAR SIDE
59	40887	UNIVERSITY AV (HWY 47)	53 AV NE	53UV	FAR SIDE
60	40226	UNIVERSITY AV (HWY 47)	57 AV NE		FAR SIDE
61	40228	UNIVERSITY AV (HWY 47)	61 AV NE		FAR SIDE
62	40235	UNIVERSITY AV (HWY 47)	MISSISSIPPI ST	MIUV	FAR SIDE
63	40898	UNIVERSITY AV NE	MISSISSIPPI / 69 AV		MID BLOCK
64	40230	UNIVERSITY AV (HWY 47)	69 AV NE		FAR SIDE
65	40232	UNIVERSITY AV (HWY 47)	73 AV NE	73UV	FAR SIDE
66	49280	UNIVERSITY AV (HWY 47)	OSBORNE RD NE		FAR SIDE
67	49383	UNIVERSITY AV (HWY 47)	81 AV NE		FAR SIDE
68	42381	NORTHTOWN TRANSIT	LOCAL STOP	NOTW	NEAR SIDE

Table D.3. List of route #10 SB stops

Stop_seq	Site_ID	Site_on	Site_at	Node_ID	Corner_desc
1	42381	NORTHTOWN TRANSIT	LOCAL STOP	NOTW	NEAR SIDE
2	49384	UNIVERSITY AV (HWY 47)	81 AV NE		FAR SIDE
3	40237	UNIVERSITY AV (HWY 47)	OSBORNE RD NE		FAR SIDE
4	40234	UNIVERSITY AV (HWY 47)	73 AV NE	73UV	FAR SIDE
5	40231	UNIVERSITY AV (HWY 47)	69 AV NE		FAR SIDE
6	40889	UNIVERSITY AV NE	69 AV / MISSISSIPPI		MID BLOCK
7	40236	UNIVERSITY AV (HWY 47)	MISSISSIPPI ST	MIUV	FAR SIDE
8	42212	UNIVERSITY AV (HWY 47)	SATELLITE LANE		FAR SIDE
9	40229	UNIVERSITY AV (HWY 47)	61 AV NE		FAR SIDE
10	40227	UNIVERSITY AV (HWY 47)	57 AV NE		FAR SIDE
11	14182	53 AV NE	UNIVERSITY AV SVC RD	53UV	FAR SIDE
12	14177	53 AV NE	5 ST NE		NEAR SIDE
13	14173	53 AV NE	7 ST		NEAR SIDE
14	14169	53 AV NE	SULLIVAN DR		NEAR SIDE
15	14166	53 AV NE	MONROE ST NE		ACROSS FROM
16	14168	53 AV NE	TARGET - E ENTRANCE		ACROSS FROM
17	17138	CENTRAL AV NE	53 AV NE	53CE	FAR SIDE
18	17141	CENTRAL AV NE	52 AV NE		FAR SIDE
19	17142	CENTRAL AV NE	51 AV NE	51CE	NEAR SIDE
20	17145	CENTRAL AV NE	50 AV NE		NEAR SIDE
21	17146	CENTRAL AV NE	49 AV NE		FAR SIDE
22	17149	CENTRAL AV NE	48 AV NE		FAR SIDE
23	17150	CENTRAL AV NE	47 AV NE		ACROSS FROM
24	17153	CENTRAL AV NE	46 AV NE		FAR SIDE
25	17154	CENTRAL AV NE	45 AV NE		FAR SIDE
26	17157	CENTRAL AV NE	44 AV NE		NEAR SIDE
27	17158	CENTRAL AV NE	43 AV NE		NEAR SIDE
28	17161	CENTRAL AV NE	42 AV NE		NEAR SIDE

29	17162	CENTRAL AV NE	41 AV NE	41CE	NEAR SIDE
30	17165	CENTRAL AV NE	40 AV NE	4 CE	NEAR SIDE
31	17166	CENTRAL AV NE	39 AV NE		NEAR SIDE
32	17170	CENTRAL AV NE	37 AV NE		NEAR SIDE
33	17171	CENTRAL AV NE	COLUMBIA PKWY		NEAR SIDE
34	17175	CENTRAL AV NE	35 AV NE		ACROSS FROM
35	17179	CENTRAL AV NE	33 AV NE		ACROSS FROM
36	17180	CENTRAL AV NE	ST ANTHONY PKWY		FAR SIDE
37	17183	CENTRAL AV NE	31 AV NE		ACROSS FROM
38	17187	CENTRAL AV NE	29 AV NE		ACROSS FROM
39	17188	CENTRAL AV NE	28 AV / 27 AV NE		MID BLOCK
40	17195	CENTRAL AV NE	LOWRY AV NE	LWCE	NEAR SIDE
41	17196	CENTRAL AV NE	24 AV NE		NEAR SIDE
42	17200	CENTRAL AV NE	22 AV NE		NEAR SIDE
43	17204	CENTRAL AV NE	19 AV NE		NEAR SIDE
44	17207	CENTRAL AV NE	18 1/2 AV NE		NEAR SIDE
45	17208	CENTRAL AV NE	18 AV NE		NEAR SIDE
46	17212	CENTRAL AV NE	14 AV NE		NEAR SIDE
47	17216	CENTRAL AV NE	BROADWAY ST		FAR SIDE
48	17220	CENTRAL AV NE	SPRING ST		FAR SIDE
49	17225	CENTRAL AV NE	7 ST SE		NEAR SIDE
50	17227	CENTRAL AV NE	HENNEPIN AV E		NEAR SIDE
51	46801	CENTRAL AV	4 ST SE	CEUN	FAR SIDE
52	12282	CENTRAL AV	2 ST SE		ACROSS FROM
53	19270	3 AV S	1 ST / 2 ST S		MID BLOCK
54	19308	WASHINGTON AV S	3 AV / 2 AV S	1S2A	MID BLOCK
55	17976	NICOLLET MALL	3 ST S	3SNI	NEAR SIDE
56	52320	NICOLLET MALL	4 ST S		NEAR SIDE
57	17978	NICOLLET MALL	5 ST S		NEAR SIDE
58	17979	NICOLLET MALL	6 ST S		NEAR SIDE
59	17980	NICOLLET MALL	7 ST S	7SMA	NEAR SIDE
60	17981	NICOLLET MALL	8 ST S	8SNI	NEAR SIDE
61	17982	NICOLLET MALL	8 ST / 9 ST S		MID BLOCK
62	17983	NICOLLET MALL	9 ST / 10 ST S		MID BLOCK
63	42837	LEAMINGTON RAMP	UPPER LEVEL	LMRP	NEAR SIDE

APPENDIX E: DATA PROCESSING SQL SCRIPTS

E.1 Link Travel Time

```
IF EXISTS(SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE
TABLE_NAME = 'tp2tp_travel_time')  DROP TABLE tp2tp_travel_time
go
create table tp2tp_travel_time (
line_id varchar(8),
line_direction varchar(8),
veh_id int,
block_number smallint,
seg_from varchar(8),
seg_to varchar(8),
service_id varchar(8),
pat_id int,
t_dep_h float,
rnd_t_dep_h int,
t_arr_h float,
travel_time_min float,
rnd_travel_time_min int,
sched_travel_min float,
travel_speed_mph float)
go
insert into tp2tp_travel_time
select  a.line_id, a.line_direction, a.veh_id, a.block_number,
a.seg_to, b.seg_to, a.service_id, a.pat_id,
a.act_dep_at_tp/3600.0, round(a.act_dep_at_tp/3600.0,0),
b.act_arr_at_tp/3600.0,
(b.act_arr_at_tp-a.act_arr_at_tp)/60.0,
round((b.act_arr_at_tp-a.act_arr_at_tp)/60.0, 0),
(b.sched_time-a.sched_time)/60.0,
(b.odometer-a.odometer)*36.0/(b.act_arr_at_tp-a.act_arr_at_tp)
from t_apc_poc a
inner join t_apc_poc b
on a.line_id=b.line_id and a.line_direction=b.line_direction
and a.seg_from=b.seg_to and a.seg_from<>a.seg_to and b.seg_from<>b.seg_to
and a.trip_number=b.trip_number and a.veh_id=b.veh_id and a.pat_id = b.pat_id

where b.act_arr_at_tp>0 and a.act_arr_at_tp>0
and (b.act_arr_at_tp>a.act_arr_at_tp)
and b.odometer>a.odometer
go
```

E.2 Trip Travel Time

```
create view tmp as
select  a.line_id, a.line_direction, a.veh_id, a.block_number,
a.seg_to as seg_from, b.seg_to, a.service_id, a.pat_id,
a.act_dep_at_tp/3600.0 as hr_dep, round(a.act_dep_at_tp/3600.0,0) as hr_dep_hr,
b.act_arr_at_tp/3600.0 as hr_arr,
(b.act_arr_at_tp-a.act_arr_at_tp)/60.0 as TT,
round((b.act_arr_at_tp-a.act_arr_at_tp)/60.0, 0) as rnd_TT,
(b.sched_time-a.sched_time)/60.0 as sched_TT,
(b.odometer-a.odometer)*36.0/(b.act_arr_at_tp-a.act_arr_at_tp) as speed
from t_apc_poc a
inner join t_apc_poc b
on a.line_id=b.line_id and a.line_direction=b.line_direction
and a.node_id='CEUN' and b.node_id='51CE'
and a.trip_number=b.trip_number and a.veh_id=b.veh_id and a.pat_id = b.pat_id
where b.act_arr_at_tp>0 and a.act_arr_at_tp>0
and (b.act_arr_at_tp>a.act_arr_at_tp)
and b.odometer>a.odometer
go
select avg(TT) as ttmean,
stdev(TT) as tt_sd ,
count(*) as N
from tmp
--where veh_id>=7212 and veh_id<=7215
where (veh_id<7212 or veh_id>7215)
go
drop view tmp
go
```

APPENDIX F: SAMPLE SOURCE CODE

F.1 GPRMC

```
#include <math.h>
#include "gps.h"
#include <stdlib.h>
// GPRMC_1.c
struct schedule
{
    char service[10];
    int route;
    int trip;
    char dir[10];
    long stop_id;
    char time[5];
    int seq;
    char tmpt[10];
    char trm[5];
};

//=====================================================================
=====
// function to convert latitude/longitude from GPS format into degrees(decimal format).
//=====================================================================
=====
double lat_gps(double lat)
{
    double final_lat;
    double join_decim;
    long lat_int;
    double lat_decim;
    int sign=0;

    sign = (int)(lat/abs(lat));
    lat_int = abs((long)lat);
    lat_decim = fabs(lat) -(double)lat_int;

    join_decim = lat_decim * 60;
    if(fabs(join_decim) >= 100)
        final_lat = ((double)(lat_int*1000) + join_decim) * sign;
    else
        final_lat = ((double)(lat_int *100) + join_decim) * sign;
    return final_lat;
}

//============= structure for storing the broken GPS string tokens=============
struct GPRMC
```

```
{
    int field_id[14];
    char field_name[14][40];
    char field_val[14][15];
};

//============= structure for storing the SC commands reply tokens=============
struct SC
{
    int EnDis;
    char field_val[15];
};

//============ structure for displaying the SC structure members=============
void dispSC(struct SC *stc)
{
    printf("\n the SC_EnDis signal is ^^%s^^ and the value =%d",stc->field_val, stc->EnDis);
}

//============ structure for displaying the GPRMC struct members=============
void dispGPR(struct GPRMC str1)
{
    int i;

    printf("\nkey       value");
    for(i=0;i<14;i++)
    {
        printf("\n%d        %s                          %s",str1.field_id[i],str1.field_name[i],
str1.field_val[i]);
    }
}

//============ structure for storing the Emtrac query command reply tokens====
struct QUERY
{
    int vehi_id;
    int EnDis;
    char quer_reply[15];
};

//======== function for calculating the cartesian distance between 2 points given
//======== in cartesian (x,y)
format============================================
double distance(double a_x, double a_y, double b_x, double b_y)
{
    double sqre=0, dista=0;
```

```
//	printf("\n cordinates=====%f===%f===%f===%f", a_x,a_y,b_x,b_y);

	sqre = pow(a_x - b_x, 2)+pow(a_y - b_y,2);
	dista = sqrt(sqre);
//	printf("\n distance = %f",dista);
	return dista;
}

//=========== structure for storing the stop information=====================
struct stops
{
	int stop_seq;
	long site_id;
	char site_on[30];
	char site_at[30];
	char node_id[10];
	char corner_desc[30];
	double lati;
	double longi;
//	char tsp_stp[10];
	int t_dwell;
	double cart_x,cart_y;

	double dista;
};

//========== sturcture for storing the intersections information=============
struct inters
{
	int emtrac_id;
	char intx_on[30];
	char intx_at[30];
	double lati;
	double longi;
	int tsp;

	double cart_x,cart_y;

	double dista;
};


/* ================== NOT in use codes ================================
====================================================================
====
```

```
void empty_gpr(char (*str)[10])
{
     int i;

     for(i=0;i<14;i++)
     {
          strcpy(*(str+i),"");
     }
}
==============================================================================
====
*/

//========== sturcture for storing the GPS information in correct datatype format
//==============================================================================
=========
struct new_GPRMC
{

// member field3 is the latitude in double
// member field5 is the longitude in double

     int field1, field9;
     int hh,mm,ss;

     double tot_lat,tot_long;

     int lat_deg, long_deg;
     double lat_mins, long_mins;
     double field3, field5, field7, field8;
     char field_str0[10], field_str2[10], field_str4[10], field_str6[10], field_str10[10],
field_str11[10];
     double cartesian_x, cartesian_y;
};

//======== function for displaying the struct new_GPRMC members
====================
void disp_newGPR(struct new_GPRMC *str2)
{
//   int i;

    if(rec==1)
    {
     fp=fopen(rec_file, "a+");
     if(fp!=NULL)
     {
```

```
/*      fprintf(fp, "\nkey       value");
        fprintf(fp, "\n str0 = %s",str2->field_str0);  // @@$GPRMC text
        fprintf(fp, "\n int1 = %d",str2->field1);  // time stamp
        fprintf(fp, "\n str2 = %s",str2->field_str2);  //Validity A-valid, V-invalid
        fprintf(fp, " %f,",str2->field3);  //current latitude
        fprintf(fp, " %f,",str2->field5);  // current longitude
        fprintf(fp, " %s,",str2->field_str4);  // North/South
        fprintf(fp, " %s,",str2->field_str6);  // East/West
        fprintf(fp, " %f,",str2->field7);  // speed in knots
        fprintf(fp, " %f,",str2->field8);  // true course
        fprintf(fp, " %d,",str2->field9);  // date stamp
        fprintf(fp, " %s,",str2->field_str10);  // magnetic variation
        fprintf(fp, " %s,",str2->field_str11);  //  East/west checksum
*/
//      fprintf(fp, "\n Current position------");
        fprintf(fp, "\n %d:%d:%d,",str2->hh,str2->mm,str2->ss);
        fprintf(fp, "%s", tripdNS);
        fprintf(fp, " %f,%f, ", str2->tot_lat, str2->tot_long);
        fprintf(fp, " %f,%f", str2->cartesian_x, str2->cartesian_y);


//      fprintf(fp, " %s-%s, %f", NorS,EorW,str2->field7);


        fclose(fp);
    }
   }

/*  printf("\nkey       value");
//  printf("\n str0 = %s",str2->field_str0);
//  printf("\n int1 = %d",str2->field1);
//  printf("\n str2 = %s",str2->field_str2);
    printf("\n int3 = %f",str2->field3);
    printf("\n str4 = %s",str2->field_str4);
    printf("\n int5 = %f",str2->field5);
    printf("\n str6 = %s",str2->field_str6);
    printf("\n int7 = %f",str2->field7);
//  printf("\n int8 = %f",str2->field8);
//  printf("\n int9 = %d",str2->field9);
//  printf("\n str10 = %s",str2->field_str10);
    printf("\n str11 = %s\n",str2->field_str11);
*/
    printf("\n Current position------");
    printf(" (%f,%f)----(%f,%f)", str2->cartesian_x, str2->cartesian_y, str2->tot_lat, str2-
>tot_long);
```

```
    printf("\n Time ----- HH:MM:SS = %d:%d:%d",str2->hh,str2->mm,str2->ss);

    printf("\n Direction = %s-%s \t------\t Speed = %f \t -- trip dir = %s", NorS,EorW,str2-
>field7, tripdNS);
}


/***************************************************************************
*This function converts the Latitude/Longitude GPS measurement to the
* Coordinate System (Cartesian) designated by COORDINATE_SYS.  X points
* to the East and Y points to the North.  The units are in meters.
***************************************************************************/
/* ======= function to convert the position in (lati,longi) format to cartesian
        coordinates (x,y) format...  The units are in meters..
=============================================================================
==*/
void convert_to_cartesian1(double latDeg, double longDeg, double *cartX, double *cartY)
{
  double sin_phi0, sin_phi, deg2rad ;
  double gamma, Ln1, Ln2, R, Q ;
  coor_const *constants ;

  constants = (coor_const*)malloc(sizeof(coor_const)) ;
  init_coor_constants(constants) ;

  sin_phi0 = sin((constants->phi_zero*DEG_2_RAD)) ;
  gamma = ((constants->lambda_zero)+longDeg)*sin_phi0 *DEG_2_RAD;
//   printf("\n=======%f=========",DEG_2_RAD);
  sin_phi = sin(latDeg*DEG_2_RAD) ;
//   printf("\n=======%f=========",latDeg);
//   printf("\n=======%f===========%f======%f",sin_phi0,gamma,sin_phi);

  Ln1 = log((1.0+sin_phi)/(1.0-sin_phi)) ;
  Ln2 = log((1.0+constants->E*sin_phi)/(1.0-constants->E*sin_phi));

//   printf("\n======%f==========%f",Ln1,Ln2);

  Q = 0.5*(Ln1-constants->E*Ln2);
  R = constants->K/exp(Q*sin_phi0);

//   printf("\n======%f=========%f",Q,R);

  *cartX = (constants->E_zero+R*sin(gamma));
  *cartY = (constants->Rb+constants->Nb-R*cos(gamma));

}
```

```
//================== function to convert the strings into numbers============
struct new_GPRMC* convert(struct GPRMC str1)
{
int new_hh, new_mm;
	struct new_GPRMC *str2;
	str2 = (struct new_GPRMC*)malloc(sizeof(struct new_GPRMC));
	str2->field1 = atoi(str1.field_val[3]);
	str2->field3 = strtod(str1.field_val[5],NULL);
	str2->field5 = strtod(str1.field_val[7],NULL);
	str2->field7 = strtod(str1.field_val[9],NULL);
	str2->field8 = strtod(str1.field_val[10], NULL);
	str2->field9 = atoi(str1.field_val[11]);

	strcpy(str2->field_str0, str1.field_val[2]);
	strcpy(str2->field_str2, str1.field_val[4]);
	strcpy(str2->field_str4, str1.field_val[6]);
	strcpy(str2->field_str6, str1.field_val[8]);
	strcpy(str2->field_str10, str1.field_val[12]);
	strcpy(str2->field_str11, str1.field_val[13]);

// time
	str2->ss = (str2->field1)%100;
	str2->hh = (str2->field1)/10000;
	str2->mm = (((str2->field1)/100)%100);

//	new_hh = ((str2->field1)/10000) + (int)TZHH;
    if(TZHH < 0)
    {
	new_mm = (((str2->field1)/100)%100) - TZMM;
	if(new_mm < 0)
	{
		str2->mm = new_mm + 60;
		str2->hh = str2->hh + TZHH - 1;
	}
	else
	{
	str2->mm = new_mm;
	str2->hh = str2->hh + TZHH;
	}
    }
    else
    {
	new_mm = (((str2->field1)/100)%100) + TZMM;
	if(new_mm > 60)
```

```
    {
        str2->mm = new_mm - 60;
        str2->hh = str2->hh + TZHH + 1;
    }
    else
    {
        str2->mm = new_mm;
        str2->hh = str2->hh + TZHH;
    }
   }

//==============================================

// latitude
    str2->lat_mins = (fmod((str2->field3),(double)100))/60;
    str2->lat_deg = (str2->field3)/100;

// longitude
    str2->long_mins = (fmod((str2->field5),(double)100))/60;
    str2->long_deg = (str2->field5)/100;

// total lati and longitude
    str2->tot_lat = str2->lat_deg + str2->lat_mins;
    if(strcmp(str2->field_str6,"W")==0)
        str2->tot_long =(-1) * (str2->long_deg + str2->long_mins);
    else
        str2->tot_long = str2->long_deg + str2->long_mins;

// one line function call to convert the latitude and longitude into cartesian
    convert_to_cartesian1(str2->tot_lat, str2->tot_long, &str2->cartesian_x, &str2->cartesian_y);
//====================================================================

//  free(str2);
    return(str2);
}

/*==================NOT in use codes================================
void empty_newgpr(struct new_GPRMC *new_gpr)
{
    new_gpr->field1 = -1;
    new_gpr->field9 = -1;
    new_gpr->hh = 0;
    new_gpr->mm = 0;
    new_gpr->ss = 0;
    new_gpr->lat_deg = 0;
    new_gpr->long_deg = 0;
```

```
    new_gpr->lat_mins = 0;
    new_gpr->long_mins = 0;
    new_gpr->field3 = 0;
    new_gpr->field5 = 0;
    new_gpr->field7 = 0;
    new_gpr->field8 = 0;
    new_gpr->cartesian_x = 0;
    new_gpr->cartesian_y = 0;

    strcpy(new_gpr->field_str0,"");
    strcpy(new_gpr->field_str2,"");
    strcpy(new_gpr->field_str4,"");
    strcpy(new_gpr->field_str6,"");
    strcpy(new_gpr->field_str10,"");
    strcpy(new_gpr->field_str11,"");
}
======================================================================
===*/
```

F.2 Schedule

```
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/io.h>
#include <string.h>

void schedules(struct schedule *sched, long dum_stop, int hrs, int mins)
{
sched = (struct schedule*)malloc(sizeof(struct schedule));

    FILE *ifp;
    int i,j;
    int read_length=80;
    char line[read_length];
    char *brk;
    char ndirec[20],nstop[10],ntime[10], ntrip[10];
    int pdirec,pstop,ptime, ptrip;
    int *lines;
    char *vdirec;
    char vtime[10];
```

```
    int temp_trip;
    int findtrip;
    int crryon;
    int rev=0;
    long vstop;

    int line_num;
    int finish = 0;
    int end=0, out=0;
    int match_line;

    int interval = 0;
    int interval1 = 0;
    int LtEar=0;

    char *split;
    int done =0;
    int hh,mm;
    char file_name[90]="/home/varun/Documents/fall-
10/work/1_dec/sched_dir/Rte10Sched_WK";
    sprintf(file_name,"%s_%ld.csv", file_name, dum_stop);

    ifp = fopen(file_name ,"r");


    if (ifp == NULL)
    {
        printf("\n Can't open specific schedule file");
       if(rec==1)
       {
         fp = fopen(rec_file, "a+");
         if(fp != NULL)
         {
            fclose(fp);
         }
       }
        if(prev_sched_mm > 0)
        {
                printf("\n 3.. the trip number is: %d", tripn);
                printf("\n prevsly the bus is late by %d mins", prev_sched_mm);
          if(rec==1)
          {
                fp = fopen(rec_file, "a+");
            if(fp != NULL)
                {
```

```
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "%d,", prev_sched_mm);
                fclose(fp);
                }
           }
        }
        else if(prev_sched_mm < 0)
        {
                printf("\n 4.. the trip number is: %d", tripn);
                printf("\n prevsly the bus is early by %d mins", prev_sched_mm);
           if(rec==1)
           {
                fp = fopen(rec_file, "a+");
                if(fp != NULL)
                {
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "%d", prev_sched_mm);
                        fclose(fp);
                }
           }
        }
        else if(prev_sched_mm==0)
        {
                printf("\n 5.. the trip number is: %d", tripn);
                printf("\n prevsly the bus is on time");
            if(rec==1)
            {
              fp = fopen(rec_file, "a+");
              if(fp != NULL)
                {
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "0");
                fclose(fp);
                }
            }
        }
    }
    else
    {
        line_num=0;

           if(!feof(ifp) && line_num==0)
           {
                fgets(line, read_length, ifp); // read a line
                line[strlen(line)-1]='\0';
```

```
		i = 0;
		brk = strtok(line,",");

		while(brk != NULL)
		{
			if(strstr(brk,"Di") != NULL)
		      {
			strcpy(ndirec,brk);
			    pdirec = i;
		      }
			else if(strstr(brk,"stop") != NULL || strstr(brk,"Stop") != NULL)
			{
				strcpy(nstop,brk);
				pstop = i;
			}
			else if(strstr(brk,"time") != NULL || strstr(brk,"Time") != NULL)
			{
				strcpy(ntime,brk);
				ptime = i;
			}
			else if(strstr(brk, "trip") != NULL || strstr(brk, "Trip") != NULL)
			{
				strcpy(ntrip, brk);
				ptrip = i;
			}
			i++;
			brk = strtok(NULL,",");
		}
		line_num++;
	}

	   while(!feof(ifp) && out==0)
	   {
		fgets(line, read_length, ifp); // read a line
		line[strlen(line)-1]='\0';

		i = 0;
		finish = 0; crryon = 0; findtrip = 0;
		brk = strtok(line,",");
		while(brk != NULL)
		{
		   if(i == ptrip)
		   {
		      temp_trip = atoi(brk);
			if(trip_ch == 0)
			{
```

```
            if(tripn != 0)
            {
                if(temp_trip == tripn)
                {
                        crryon = 1;
                }
            }
        }
        else
        {
            findtrip = 1;
        }
}
else if(i==pstop)
{
  if(crryon==1 || findtrip==1)
  {
        if(atoi(brk) == dum_stop)
        {
            vstop = atoi(brk);
            finish = 1;
        }
  }
}
else if(i==ptime && finish==1)
{
        strcpy(vtime,brk);
        j = 0;done=0;
        split = strtok(brk,":");
        while(split != NULL)
        {
           if(j==0)
           {
               hh = atoi(split);
               if(hh==hrs)
               {
                       done = 1;
               }
               else if(hh==hrs+1)
               {
                       done = 2;
               }
           }
           if(j==1 && done==1)
           {
                   mm = atoi(split);
```

```
					if(findtrip==1)
					{
						if(mm > mins)
						{
							out=1;
							if(prev_mm > mins)
								interval1 = mins+60-prev_mm;
							else
								interval1 = mins - prev_mm;
							if(prev_mm > mm)
								interval = -prev_mm +60 + mm;
							else
								interval = mm - prev_mm;
	printf("\n ---1... interval = %d -- interval1 = %d -- prev_mm = %d", interval, interval1,
prev_mm);
							if(interval1 <= 0.8*interval)
							{
								match_line = line_num-1;
								if(mins < prev_mm)
									LtEar = mins + 60 - prev_mm;
								else
									LtEar = mins - prev_mm;
							}
							else
							{
								match_line = line_num;
								LtEar = mm - mins;
							}
	printf("\ntime.. sched.. the match line = %d", match_line);
						}
					}
					else if(crryon==1)
					{
						out = 1;
						match_line= line_num;
						LtEar = mins - mm;
					}
						prev_mm = mm;
						prev_sched_mm = LtEar;
						prev_sched_stop_id = vstop;
						strcpy(prev_sched_time,vtime);
				}
				else if(j==1 && done==2)
				{
						mm = atoi(split);
						if(findtrip==1)
```

```
                        {
                                out = 1;
                                interval1 = mins + 60 -prev_mm;
                                interval = mm + 60 - prev_mm;
    printf("\n ---2... interval = %d -- interval1 = %d -- prev_mm = %d", interval, interval1,
prev_mm);
                                if(interval1 <= 0.8*interval)
                                {
                                        match_line = line_num-1;
                                        if(mins < prev_mm)
                                                LtEar = mins +60- prev_mm;
                                        else
                                                LtEar = mins - prev_mm;
                                }
                                else
                                {
                                        match_line = line_num;
                                        if(mins < mm)
                                                LtEar = mm +60- mins;
                                        else
                                                LtEar = mm - mins;
                                }
    printf("\ntime.. sched..  the match line = %d", match_line);
                        }
                        else if(crryon==1)
                        {
                                out = 1;
                                match_line = line_num;
    printf("\n the match line = %d", match_line);
                        LtEar = -1*((60-mins)+mm);
                        }
                                prev_mm = mm;
                        prev_sched_mm = LtEar;
                        prev_sched_stop_id = vstop;
                        strcpy(prev_sched_time,vtime);
                    }
                    else if(j==1)
                    {
                        mm = atoi(split);
                        prev_mm = mm;

                    }

                    split = strtok(NULL,":");
                }
```

```
                }
                i++;
                brk = strtok(NULL,",");
              }
              line_num++;
        }
        fclose(ifp);

       if(out != 1)
       {
        if(prev_sched_mm > 0)
        {
                printf("\n 3.. the trip number is: %d", tripn);
                printf("\n prevsly the bus is late by %d mins", prev_sched_mm);
           if(rec==1)
           {
                fp = fopen(rec_file, "a+");
              if(fp != NULL)
                {
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, " %d,", prev_sched_mm);
                fclose(fp);
                }
           }
        }
        else if(prev_sched_mm < 0)
        {
                printf("\n 4.. the trip number is: %d", tripn);
                printf("\n prevsly the bus is early by %d mins", prev_sched_mm);
           if(rec==1)
           {
                fp = fopen(rec_file, "a+");
                if(fp != NULL)
                {
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "%d,", prev_sched_mm);
                        fclose(fp);
                }
           }
        }
        else if(prev_sched_mm==0)
        {
//              printf("\n the value of time: %s and value of stop is %d", prev_sched_time,
prev_sched_stop_id);
                printf("\n 5.. the trip number is: %d", tripn);
                printf("\n prevsly the bus is on time");
```

```
            if(rec==1)
            {
              fp = fopen(rec_file, "a+");
              if(fp != NULL)
                  {
                  fprintf(fp, ", %d,", tripn);
                          fprintf(fp, "0");
                  fclose(fp);
                  }
            }
        }
      }

    if(out==1)
    {
     ifp = fopen(file_name,"r");

     if (ifp == NULL)
     {
         printf("Can't open input file in.list!\n");
          if(rec==1)
          {
           fp = fopen(rec_file, "a+");
           if(fp != NULL)
           {
             fprintf(fp, ", Can't open input file in.list! ,");
               fclose(fp);
           }
         exit(1);
        }
     }
     else
     {
         line_num=0;
         while(!feof(ifp))
         {
              fgets(line, read_length, ifp); // read a line
                line[strlen(line)-1]='\0';
            if(line_num == match_line)
            {
                   i=0;
                   brk=strtok(line,",");
                   while(brk != NULL)
                   {
                          if(i==0)
                          {
```

```
				strcpy(sched->service,brk);
			}
			else if(i==1)
			{
				sched->route = atoi(brk);
			}
			else if(i==2)
			{
				sched->trip = atoi(brk);
					tripn = sched->trip;
				if(findtrip == 1)
				{
					tripn = sched->trip;
					trip_ch = 0;
					findtrip = 0;
				}
			}
			else if(i==3)
			{
				strcpy(sched->dir,brk);
			}
			else if(i==4)
			{
				sched->stop_id = atoi(brk);
			}
			else if(i==5)
			{
				strcpy(sched->time,brk);
			}
			else if(i==6)
			{
				sched->seq = atoi(brk);
			}
			else if(i==7)
			{
				strcpy(sched->tmpt,brk);
			}
			else if(i==8)
			{
				strcpy(sched->trm,brk);
			}
			i++;
			brk = strtok(NULL, ",");
		}
	}
	line_num++;
```

```
        }
    }
    fclose(ifp);

    if(prev_sched_mm > 0)
    {
        printf("\n 6.. the trip number is: %d", tripn);
        printf("\n the bus is late by %d mins", prev_sched_mm);
        if(rec==1)
        {
                fp = fopen(rec_file, "a+");
            if(fp != NULL)
                {
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "%d,", prev_sched_mm);
                fclose(fp);
                }
        }
    }
    else if(prev_sched_mm < 0)
    {
        printf("\n 7..  the trip number is: %d", tripn);
        printf("\n the bus is early by %d mins", prev_sched_mm);
         if(rec==1)
        {
                fp = fopen(rec_file, "a+");
                if(fp != NULL)
                {
                fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "%d,", prev_sched_mm);
                        fclose(fp);
                }
        }
    }
    else if(prev_sched_mm==0)
    {
        printf("\n 8.. the trip number is: %d", tripn);
        printf("\n the bus is on time");
        if(rec==1)
        {
            fp = fopen(rec_file, "a+");
            if(fp != NULL)
                {
                        fprintf(fp, ", %d,", tripn);
                        fprintf(fp, "0");
                fclose(fp);
```

```
                    }
               }
            }

           }
         }
       }
```