CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Development of Hardware in the Loop Simulation and Paramics/VS-PLUS Integration

Susan Dickey, et al.

California PATH Research Report
UCB-ITS-PRR-2008-29

Final Report for Task Order 5311

November 2008

ISSN 1055-1425

Caltrans Task Order 5311:

Development of Hardware in the Loop Simulation and Paramics/VS-PLUS Integration

Final Report

Susan Dickey, Meng Li, Jonathan Yee, Marco Zennaro
California PATH, University of California, Berkeley

Henry X. Liu and Wenteng Ma
Department of Civil Engineering, University of Minnesota - Twin Cities

Hongchao Liu and Shuaiyu Chen
Multidisciplinary Research Center in Transportation, Texas Tech University

Wei-hua Lin and Lefei-Li
University of Arizona

Acknowledgment

This is the final report of the project “Development of Hardware-in-the-Loop (HiL) Simulation and Paramics/VS-PLUS Integration”, which was sponsored by the Department of Transportation of California (Caltrans) under Task Order 5311. Researchers at California Partners for Advanced Transit and Highways (PATH) program, University of California, Berkeley, at the Multidisciplinary Research Center in Transportation, Texas Tech University (TechMRT), at the University of Arizona and at the University of Minnesota collaborated on this report. Alexander Skabardonis served as the Principal Investigator. The contents of the report do not necessarily reflect the official views or policies of the State of California.

The authors would like to thank Jerry Kwong of the Caltrans Division of Research and Innovation (DRI), and Kai Leung and Steve Hague of the Division of Traffic Operations, Caltrans Headquarters for their support and advice during the project.

Abstract

The report describes three research efforts carried out under a project titled “Development of Hardware-in-the-Loop (HiL) Simulation and Paramics/VS-PLUS Integration” sponsored by the California Department of Transportation (Caltrans) under Task Order 5311. The first effort developed and evaluated traffic signal optimization with Hardware-in-the-Loop Simulation (HiLS), using the NIATT Controller Interface Device (CID) manufactured by McCain Traffic Supply to provide real-time linkage between the Paramics microscopic simulation and a NEMA TS1 controller. An adaptive control system incorporated the traffic flow prediction model to predict the traffic flows from the surrounding intersections, and an online signal optimization model was used to obtain the signal timing plan for the subsequent cycle, based on the traffic flows predicted in the previous cycle. The performance of the proposed adaptive control system was evaluated through a case study in which HiLS is applied to a small urban network in Logan, Utah. The second effort developed a Paramics plug-in that worked over a serial port connection to a specially modified 170 for HiL operation. After this initial development, the NIATT/McCain CID was configured to work as a Paramics plug-in with both 170 and 2070 controllers, and experiments were carried out to compare the performance of Paramics simulations with the UC Irvine Paramics signal controller plug-in with HiLS. In the third effort, investigation and evaluation of integrated Paramics/VS-PLUS software was carried out, resulting in a user’s guide for use of the integrated Paramics/VS-PLUS simulation software.

Keywords: Advanced Traffic Management Systems, Computer Simulation, Traffic Control, Traffic Flow, Traffic Signals

Executive Summary

With the continuing increase in urbanization and traffic congestion, traffic engineers have been forced to develop new traffic control systems in order to maintain safety and meet the growing transportation needs of cities. This project explored several different methods of increasing accuracy and performance in traffic control system simulation, and using simulation to optimize traffic control systems. Methods examined include Hardware-in-the-Loop Simulation (HiLS), the use of plug-in traffic signal controller modules more feature-rich than the default Paramics traffic signal simulation, and the use of the VS-PLUS traffic signal controller software, which provides software emulation of the traffic signal controller which is based on the actual code running in the controller. Results of the research can be summarized as follows:

1. Hardware-in-the Loop Simulation using the McCain Traffic Supply/NIATT Controller Interface Device (CID) is a useful tool for algorithm test and verification. The researchers at Minnesota and Utah used HiLS communicating through a real-time linkage to develop a traffic flow prediction model. Researchers at California PATH have continued to use it for lab testing of developed control systems in on-going work for Task Order 6407 "Relieve Congestion and Conflicts Between Railroad and Light Rail Grade-Crossing Intersections" and Task Order 6400 "Field Operational Tests of Adaptive Transit Signal Proprity (ATSP)". HiLS provides an excellent testing environment to test and evaluate the implementation of different traffic control system features.
2. Through limited, but reasonable and valuable testing, the researchers at Minnesota concluded that adaptive signal control performed better during periods of high demand compared to actuated signal control and pre-timed signal control. Though the actuated signal control performed better when demand was low, the inclusion of a dynamic signal optimization model and a turn count estimation model may strengthen the performance of adaptive signal control at lower demand.
3. Real-time linkage to Paramics simulations using the CID can be done with 170 and 2070 controllers. The degree to which this slows down the simulations makes it impractical to use HiLS for each controller in a large traffic signal network. However, simulations using the UC Irvine Paramics Signal Controller plug-in, which is a more faithful simulation of actual traffic signal behavior than the default Paramics traffic signal control model, show behavior similar to 2070 HiLS.
4. VS-PLUS, traffic signal controller software that has been widely used in Europe, has an implementation for the 2070 which includes a traffic signal controller emulator and Paramics integration. A user's guide to this software is provided in this report. The integrated Paramics/VS-PLUS model provides many useful functions for emulating traffic signals in traffic simulation, especially actuated signals. However, VS-PLUS is a parameter-driven program with a different programming model than U.S. traffic engineers are accustomed to using. While their product includes a "NEMA wizard" to allow intersections to be set up in a more familiar way, at the time of this research the NEMA interface had bugs which made it impossible to compare Paramics/VS-PLUS simulations directly to HiLS of 170/2070 controllers with matching configurations.

Further research should include Paramics simulation modules for traffic signal controllers that are software-identical to the controller software. Access to the source code of the software running on the traffic signal controller is required to develop such modules.

Table of Contents

List of Tables and Figures

Development of Hardware in the Loop Simulation and Paramics/VS-PLUS Integration: Final Report

1. Introduction

With the continuing increase in urbanization and traffic congestion, traffic engineers have been forced to develop new traffic control systems in order to maintain safety and meet the growing transportation needs of cities. Before these new advanced traffic systems can be deployed they require extensive and careful testing. Hardware-in-the-Loop Simulation (HiLS) is a valuable tool that has certain advantages over strictly software simulation in the development of new systems. By connecting the actual device that will be used in the field to a microscopic traffic simulation, new methods of traffic control can be tested for both for performance and for design flaws before they are deployed.

Paramics is a powerful microscopic traffic simulation tool that is being used extensively by Caltrans engineers in various traffic management and operation projects. A major drawback of Paramics lies in its over-simplified control logic of traffic signals. Only limited functions for fixed-time signal control are provided with the current version. This limitation has restricted its application in simulating real-life signal control systems.

Although it is difficult to model real signal controllers in microscopic simulation, several approaches hold great potential in this regard. This report consists of three self-contained sections describing distinct but complementary research efforts that were carried out as part of a project titled “Development of Hardware-in-the-Loop (HiL) Simulation and Paramics/VS-PLUS Integration” sponsored by the California Department of Transportation (Caltrans) under Task Order 5311. Conclusions and ideas for future work are included at the end of each section.

Section 2 describes research carried out by Henry Liu and Wenteng Ma at Utah State University and the University of Minnesota. They developed and evaluated traffic signal optimization with Hardware-in-the-Loop Simulation (HiLS), using the NIATT Controller Interface Device (CID) manufactured by McCain Traffic Supply to provide real-time linkage between the Paramics microscopic simulation and a NEMA TS1 controller. An adaptive control system incorporated the traffic flow prediction model to predict the traffic flows from the surrounding intersections, and an online signal optimization model was used to obtain the signal timing plan for the subsequent cycle, based on the traffic flows predicted in the previous cycle. The performance of the proposed adaptive control system was evaluated through a case study in which HiLS is applied to a small urban network in Logan, Utah. This section includes a literature review and substantive results on the performance of adaptive traffic signal control algorithms.

Section 3 describes work carried out by graduate students at UC Berkeley under the supervision of Alex Skabordonis and with the support of California PATH Software Functional Manager Susan Dickey. Researchers at PATH first developed a Paramics plug-in that worked over a serial port connection to a specially modified 170 for HiL operation, and carried out preliminary investigations on the possibility of using NTCIP or native Traffic Signal Control Program software for simulation. After this initial development, the NIATT/McCain CID was configured to work with a Paramics plug-in for both 170 and 2070 controllers, and experiments were carried out to compare the performance of Paramics simulations using the UC Irvine Paramics signal controller plug-in with HiLS. An effort was also made to compare the performance with VS-PLUS, but this could not be completed because of a bug in the VS-PLUS NEMA Paramics interface.

Section 4 is the results of the investigation and evaluation of the integrated Paramics/VS-PLUS software that conducted by the Multidisciplinary Research Center in Transportation at Texas Tech University (TechMRT) in collaboration with the University of Arizona. This section documents the research findings from this project in the form of a user’s guideline for easy use of the integrated Paramics/VS-PLUS simulation software. Appendix A has additional information obtained in conversation with VS-PLUS support personnel.

2. Development & Evaluation of Traffic Signal Optimization Model with Hardware-in-the-Loop Simulation

2.1 Background

In recent years, there has been a significant development in the field of electronics and computer engineering. The increase in computational power, advanced communication networks and reliable sensor technologies has resulted in the development of Intelligent Transportation Systems (ITS), and presents an opportunity for the improvements of safety and efficiency in traffic control systems (Head et al., 1992).

In the early 1960's computer-based traffic signal control was introduced and numerous experiments were conducted to develop advanced traffic responsive control strategies. As a result of these experiments the Urban Traffic Control System (UTCS) program funded by Federal Highway Administration (FHWA) came into existence. UTCS is a centralized system that controls all the signalized intersections in a network with a variety of signal timing plans. Research in the UTCS program is divided into three generations, namely First-Generation Control (1-GC), Second-Generation Control (2-GC) and Third-Generation Control (3-GC). The 3-GC is the most advanced traffic-responsive and online control system. Some of the prominent Adaptive Traffic Control System (ATCS) in the world today are SCATS (Sydney Coordinated Adaptive Traffic System) (Lowrie, 1982), SCOOTS (Split Cycle Offset Optimization Technique) (Hunt et al., 1982), PRODYN (Henry and Farges, 1989), RHODES (Real-time Hierarchical, Optimized, Distributed, and Effective System) (Head, 1995) and OPAC (Optimized Policies for Adaptive Control) (Gartner, 1990).

Adaptive signal systems need to be tested before implementation. Microscopic simulation is particularly suited for the development, testing and evaluation of intelligent transportation systems. To utilize features of field signal controllers that are not available in the traffic simulation models, new concepts such as Hardware-in-the-Loop Simulation (HiLS) are being introduced to make microscopic simulation more realistic and credible. Hardware-in-the-Loop-Simulation refers to a system in which part of a pure simulation is replaced with an actual physical component.

In this project we present an adaptive traffic control system for an isolated intersection with traffic flow prediction and signal optimization module. To evaluate the performance of the proposed adaptive control system a testing environment is developed with Hardware-in-the-Loop Simulation.

2.1.1 Problem Statement

The current signalization practice for timing intersections involves data collection during the peak hour of a typical day of the week. From the data collected, a signal timing plan is designed using an off-line computerized signal optimization program. This practice of designing an intersection timing plan does not consider variations in the traffic flow pattern. Though actuated and semi-actuated traffic signal control logic can respond to volume fluctuations, they cannot respond to changes in traffic pattern or overall increase in traffic volume. Due to the continuing growth in travel demand and change in traffic patterns, signal timing plans become outdated in the course of time. The deficiencies in current signalization practice have been overcome with real-time adaptive traffic response controls. The proposed adaptive control system predicts traffic flows based on the traffic states of the surrounding intersection and also incorporates an online signal optimization model to effectively allocate green time based on the traffic flow prediction module.

Field studies for testing and evaluating different traffic control strategies are not always preferable. Cost, safety and time duration aspects in conducting these studies have made field studies less attractive compared to microscopic traffic simulation. For present day traffic engineers and analysts, the use of microscopic simulation is an important tool for evaluation of different types of intersection controls. Since the internal software controller in the current microscopic simulation models lacks the versatility and the advanced features available in the actual traffic controller, the Hardware-in-the-Loop concept is used to bridge the gap between real world traffic operation and simulation. Hardware-in-the-Loop Simulation increases the realism of the simulation and provides accessibility to the advanced traffic controller features. The work utilizes Hardware-in-the-Loop Simulation as a testing environment to evaluate the performance of different traffic controls.

2.1.2 Overall System Architecture

Figure 2.1 represents the overall system architecture of the proposed system. The overall system architecture of the proposed system consists of three main components: traffic flow prediction, traffic signal optimization and hardware-in-the-loop simulation.

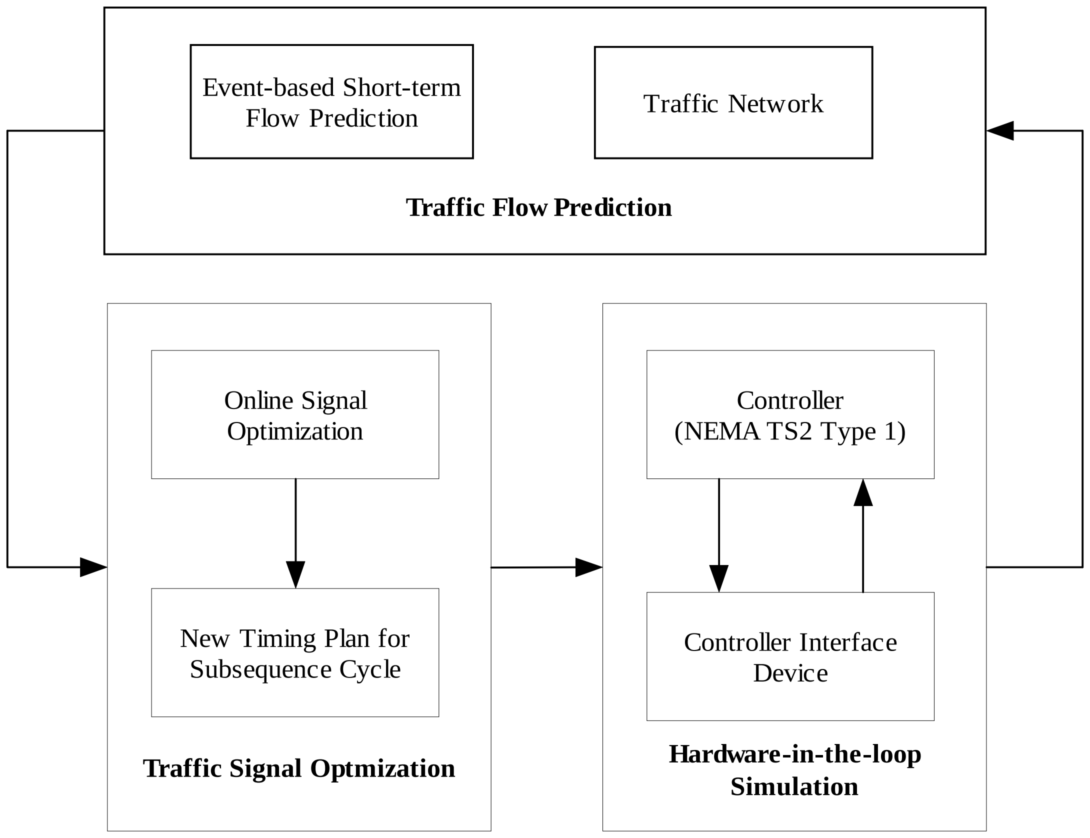

Event-based Short-term Flow Prediction
Traffic Network
Traffic Flow Prediction
Online Signal Optimization
New Timing Plan for Subsequence Cycle
Traffic Signal Optmization
Controller (NEMA TS2 Type 1)
Controller Interface Device
Hardware-in-the-loop Simulation

Figure 2.1: Overall System Architecture

A dynamic data-driven short-term traffic flow prediction model is used for predicting the traffic flows for the proposed adaptive control system. The surveillance system consists of loop detectors, which are used for the measurements for the adaptive control system. A simple online optimization model is used to develop a new signal timing plan based on the predicted traffic flows for the subsequent cycle. Variable cycle length, phase split optimization, and phase skipping are some of the key features of this signal optimization scheme. The third component of the system is to set up a testing environment to test the adaptive control strategy developed for an isolated intersection. A NEMA TS2 TYPE 1 controller (make – Econolite, model # ASC/2S – 2100) is used with CID II (Controller interface device) for the hardware-in-the-loop simulation. A simulated network in Paramics (Parallel microscopic simulation) which is based on an actual network in Logan, Utah is used for evaluation and testing of different control strategies. The proposed adaptive control system logic performs better than actuated signal control. As an outcome of this project a Paramics interface tool for HiLS through CID was developed.

2.1.3 Section Outline

This section is outlined as follows. Section 2.2 provides a brief description of the traffic flow prediction model and signal optimization model. Section 2.3 presents the concept of hardware-in-the-loop simulation. Section 2.4 describes the implementation framework of the proposed adaptive signal control. Section 2.5 evaluates the performance of the adaptive signal control via hardware-in-the-loop simulation. Section 2.6 presents the background of the proposed event-based short-term traffic flow prediction model. Section 2.7 provides the details of the green phase and red phase actuation model. Section 2.8 discusses the dynamic turning proportion prediction model. Section 2.9 evaluates the proposed traffic flow prediction model with an arterial network in Logan, UT.

2.2 Traffic Flow Prediction and Online Traffic Signal Optimization

2.2.1 Introduction

One of the key components in developing a real-time traffic adaptive signal control is traffic flow prediction. Predicting vehicle arrivals, turning probabilities and queues at an intersection is important for proactive traffic controls in order to compute phase timing that optimizes the given measure of effectiveness (e.g., control delay) (Head et al., 1992). Researchers analyzing various adaptive control systems use different flow prediction models and techniques. To build a truly real-time traffic-adaptive signal control, prediction of the temporal arrival distribution of vehicles is very important. With the information of vehicle arrivals at an intersection, the controller could decide whether to extend the green time of a phase or to terminate and switch to another phase. Timely traffic flow information is critical in optimizing and efficiently operating a signalized urban network. If the traffic flows on each link of a network are known, timing plans can be implemented to optimize coordination and effectively reduce overall vehicle delay of the system.

Most traffic response systems such as SCATS, SCOOT, and OPAC are proprietary. The research community does not know the models and algorithms implemented in these systems in detail. SCATS uses flow and occupancy data from the loop detectors that are installed on the road network for predicting traffic flows. The “strategic algorithms” in this system determine the optimum cycle length, stage splits and offsets to suit the prevailing average traffic conditions (Hunt et al., 1982). Systems such as SCATS and SCOOTS mainly depend on historical traffic data for predicting traffic flows. On the other hand, systems such as OPAC and RHODES use more advanced dynamic flow prediction techniques. The flow prediction technique used by the OPAC system is known as the rolling horizon method (Gartner, 1990). In this approach, the prediction horizon i.e. the length of the predicting period is divided into a number of small intervals. The arrival data for the beginning or the head portion of the horizon is obtained from the detectors placed upstream of each approach. The flow data for the tail or the remaining portion of the horizon is estimated from a model. The model consists of a moving average of all previous arrivals on the approach, thus the name - rolling horizon.

Inefficient and poorly designed signal timing plans result in wasted time, inefficient use of precious energy resources, environmental damage and unsafe conditions for motorists and pedestrians. The FHWA estimates that the performance of as many as 75 percent of all traffic signals could easily be improved by updating equipment, adjusting their timing plans, or by coordinating adjacent signals (Kyte et al., 2003). Traffic signal optimization models are therefore helpful in designing efficient signal timing plans, thus making intersection controls more efficient and safe. This chapter describes in detail the two traffic models, namely, traffic flow prediction and traffic signal optimization model, used in the proposed adaptive control system. The first section of this chapter deals with the traffic prediction model and the subsequent section explains the signal optimization method incorporated in the proposed adaptive control system.

2.2.2 Event-based Short-term Traffic Flow Prediction

The proposed adaptive system utilizes the event-based short-term traffic flow prediction model for predicting traffic flows. Traffic flow at the subject intersection will be predicted using detector data and signal status data from the upstream intersections. The prediction model consists of mainly four cases based on the phase status at the upstream intersection. The first two cases are when the phase status is green for the phase serving a desired movement, and the next two cases are when the phase is red for the desired movement. The factors which affect the arrival of the vehicle at the subject intersection are travel time from advanced detectors to the stop bar at the upstream intersections, delay due to an existing queue at the upstream intersections, delay due to the traffic signal at upstream intersections, travel time from the stop bar of the upstream intersection to the subject intersection and the probability that the vehicle will pass through the upstream intersections.

The travel times from the advanced detector to the stop bar and from stop bar of the upstream intersection to the subject intersection are calculated using the link flow speed and approach speed., The link flow speed and approach speed are estimated using occupancy of the conventional loop detectors and signal status of the intersection.

To estimate the delay due to the standing queue the number of vehicles in the queue should be determined. None of the current detection technology provides this as a direct traffic measurement. Therefore, the number of vehicle in the queue has to be estimated. The details of the event-based short-term traffic flow prediction model will be discussed in Section 2.6 to Section 2.9.

2.2.3 Online Traffic Signal Optimization

This section presents a detailed description of the online signal optimization model used in the proposed adaptive control system. Adaptive control systems such as SCOOT and SCATS have adapted a cyclic approach in which they update the timing plans at pre-specified time intervals. Other advanced adaptive systems like OPAC, UTOPIA, and PRODYN have dynamic signal optimization models built in them. These signal optimization algorithms are not confined to a cyclic time interval and are capable of changing the timing plan at any give time step based on the signal optimization models built in them. ALLONS-D is an example of dynamic signal optimization algorithm (Porche and Lafortune, 1997).

The online signal optimization model for the proposed adaptive system can vary the cycle length, determine phasing sequence and calculate optimal green splits for each phase. In this signal optimization model, traffic parameters, namely cycle length, phasing sequence and phase splits, are determined for the subsequent cycle instead of the projected one. Depending on the traffic flows determined by the flow prediction model, these traffic parameters can be calculated. The first step in this approach is to calculate the optimal cycle length based on Webster's equation. The second step is to determine the phasing sequence depending on the traffic flows, and the final step is to split the effective green time based on a proportional green split model (Liu et al., 2002). In the proportional green split model, the total effective green time is split between the approaches (i.e. north-south and east west) on the basis of the traffic flows and turning ratios. Each green split of the approaches is further divided proportionally between the phases. In this simple model, more green time is allocated to the more congested phase.

Compared to conventional pre-timed signal control, the proposed adaptive signal is able to adapt to both traffic flow variation and increase in traffic demand. Although actuated control logic handles the variations in traffic flow, it fails to handle the case when traffic demand increases. The signal optimization model uses the current information to determine the signal control parameters for the next cycle. Although for this project a simple signal optimization model is adapted, a more reliable dynamic approach could be incorporated in the future.

The procedure for designing the signal timing plan for the proposed adaptive control system is shown in the Figure 2.2.

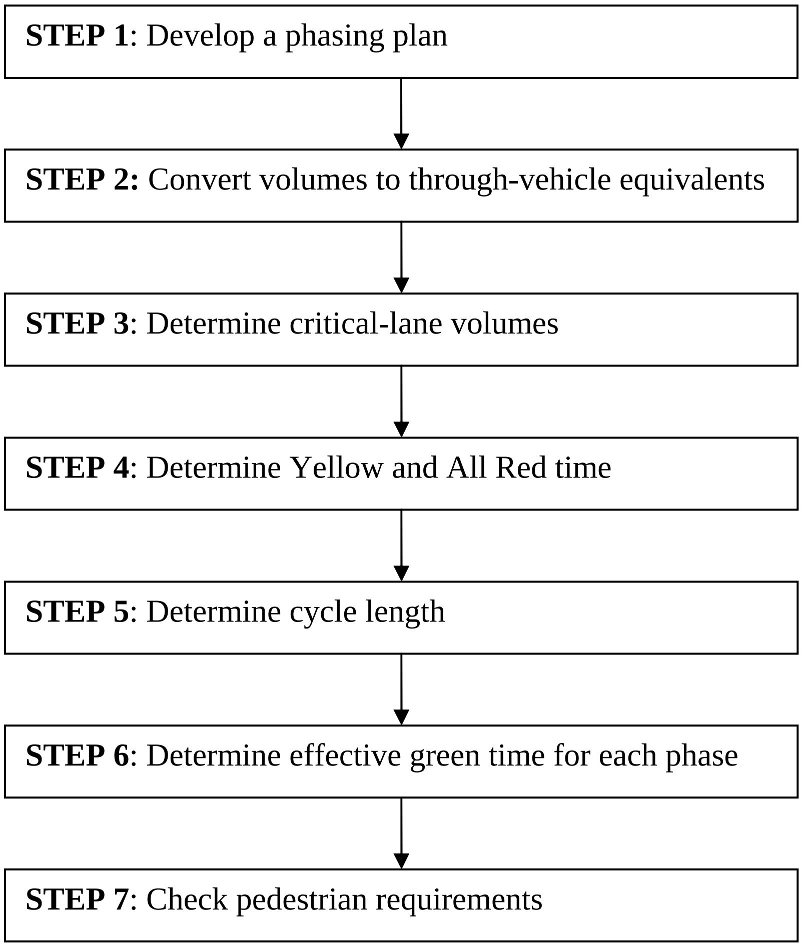

STEP 1: Develop a phasing plan
STEP 2: Convert volumes to through-vehicle equivalents
STEP 3: Determine critical-lane volumes
STEP 4: Determine Yellow and All Red time
STEP 5: Determine cycle length
STEP 6: Determine effective green time for each phase
STEP 7: Check pedestrian requirements

Figure 2.2: Traffic Signal Timing Design Procedure

The following steps describe the procedure to design the signal timing plan.

STEP 1: To develop a phase plan

Developing a phasing plan is the critical aspect of designing and timing a signal. The phasing sequence of the proposed adaptive control system is based on the dual ring phasing operation that follows the conventional NEMA (National Electrical Manufacturers Association) phase as indicated in Figure 2.2. The phasing plan consists of eight phases, but only two to six phases are implemented in any sequence. Figure 2.3 represents the phasing sequence for the proposed adaptive control system.

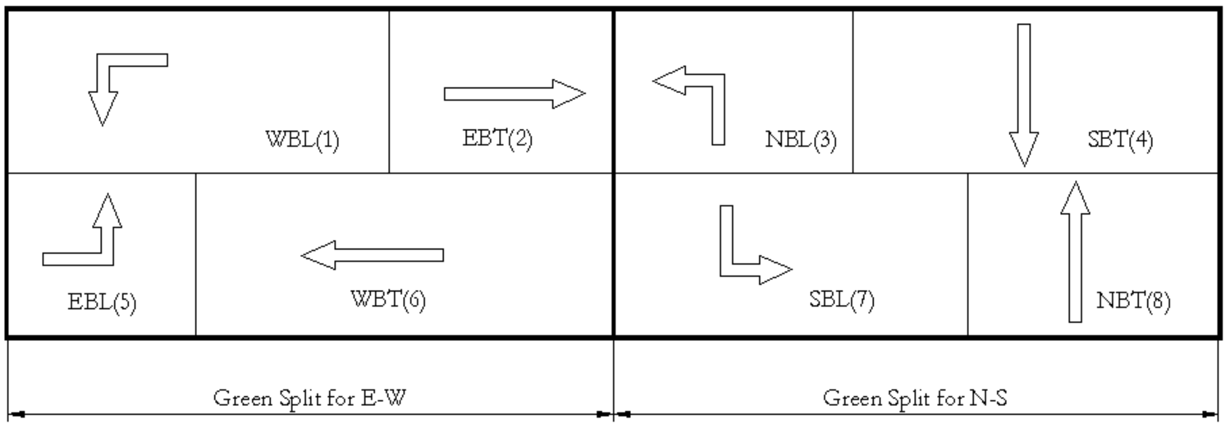


Figure 2.3: Green Split Model

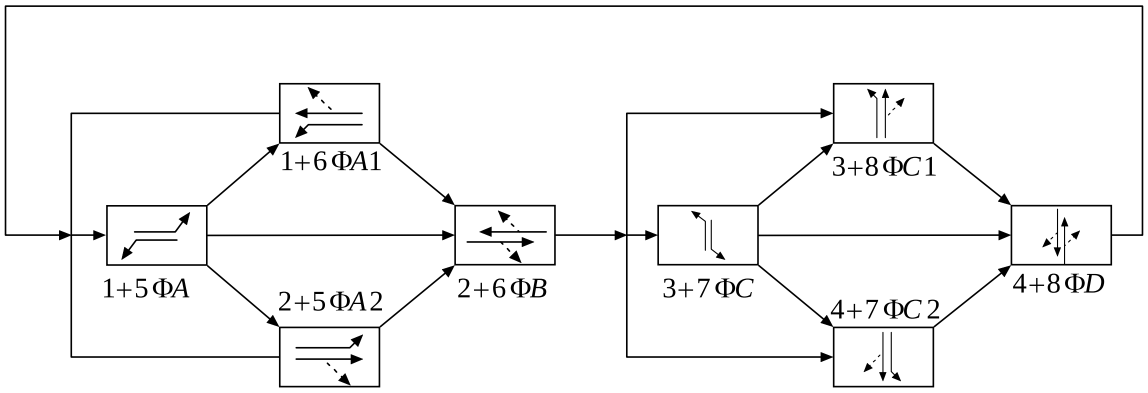


Figure 2.4: Phasing Sequence for the Proposed Adaptive Control System

STEP 2: Convert volumes to through-vehicle equivalents

The left-turn equivalency factor is determined as in section 16-12 of the Highway Capacity Manual (HCM). The left turns could be treated as permitted movements (with an opposing through flow), as protected movements (with the opposing vehicular movement stopped) or as a combination of both (compound phasing). To determine whether the left-turn movement requires a protected, permitted or combined phasing the guideline given in the HCM are followed. The guidelines in the HCM are based on two general rules namely

a. If the unadjusted left-turn volume is greater than 240 vehicles over 1 hour.

b. If the cross product of left-turn volume and the opposing through volume is greater than 50,000 for one opposing through lane, 90,000 for two opposing through lanes, or 110,000 for three or more opposing through lanes.

If one of the rules is satisfied, then a fully protected phasing for the left-turn phase is recommended. If the two-left-turning movements have different volumes on a per lane basis, then a compound phasing treatment for the left turn is recommended to reduce the delays and have a feasible cycle length.

STEP 3: Determine the critical-lane volumes

The concept of critical lane is to identify a specific lane movement which will control the timing of a given signal phase. The total signal timing is proportion to the total demand flows of the critical lanes. The following guidelines help in identifying the critical lanes (McShane et al., 2004).

a. There is a critical lane and a critical-lane flow for each discrete signal phase provided.

b. Except for lost times, when no vehicles move, there must be one and only one critical lane moving during every second of effective green time in the signal cycle.

c. Where there are overlapping phases, the potential combination of lane flows yielding the highest sum of critical lane flows while preserving the requirement of item (b) identifies critical lanes.

STEP 4: Determine the Yellow and All Red time

Change interval time (Yellow/Amber) and clearance interval time (All Red) for all the phases are calculated using Equation 2.1, Equation 2.2 and Equation 2.3.

$$y = t + \frac{1.47 S_{85}}{2a + (64.1 * 0.01G)} \tag{2.1}$$

$$ar = \frac{w + L}{1.47 S_{15}} \tag{2.2}$$

for cases in which there are no pedestrians

$$ar = \max\left[\left(\frac{w + L}{1.47 S_{15}}\right), \left(\frac{P + L}{1.47 S_{15}}\right)\right] \tag{2.3}$$

for cases in which significant pedestrian traffic exists

Where:

y = length of the yellow interval, s

t = driver reaction time, s

S_{85} = 85th percentile speed of approaching vehicles, mi/h

a = deceleration rate of vehicles, ft/s^2

G = grade of approach, %

ar = length of the all-red phase, s

w = distance from the departure stop line to the far side of the farthest conflicting traffic lane, ft

P = distance from the departure stop line to the far side of the farthest conflicting crosswalk, ft

L = length of a standard vehicle, usually taken to be 18-20 ft

S_{15} = 15th percentile speed of approaching traffic, or speed limit, as appropriate, mi/h

STEP 5: Determine the desirable cycle length.

Using the Equation 2.4 the minimum cycle length is determined. The equation used to find the cycle length is based on the maximum sum of critical lane volumes established in step 3.

$$C = \frac{1.5L + 5}{(1 - \sum y_i)} \qquad \textbf{(2.4)}$$

Where:

C = cycle length, s

L = Total lost time per cycle, s

$\sum y_i$ = summation of ratio of critical lane volume to the saturation flow per phase

STEP 6: Determine effective green time for each phase

After determining the cycle length, the effective green time per phase is determined using the proportional green split model. The total effective green time in the cycle is found by deducting the total lost time per cycle from the cycle length. The total effective green time is then split among the various phases of the signal plan in proportion to the critical lane volumes of each phase.

$$g_{TOT} = C - L \qquad \textbf{(2.5)}$$

$$g_i = g_{TOT}\left(\frac{V_{ci}}{V_c}\right) \qquad \textbf{(2.6)}$$

Where:

g_{TOT} = total effective green time in the cycle

g_i = effective green time for phase i, s

V_{ci} = critical lane volume for phase or sub phase i, veh/h

V_c = sum of the critical-lane volumes, veh/h

STEP 7: Determine pedestrian signal requirements

The last step is to check the pedestrian requirements if there are any crosswalks in the intersection. The Equation 2.7 and Equation 2.8 determine the minimum pedestrian crossing time.

$$G_p = 3.2 + \left(\frac{L}{S_p} \right) + \left(2.7 * \frac{N_{ped}}{W_E} \right) \quad \textbf{(2.7)}$$

for $W_E > 10$ ft

$$G_p = 3.2 + \left(\frac{L}{S_p} \right) + \left(0.27 * \frac{N_{ped}}{W_E} \right) \quad \textbf{(2.8)}$$

for $W_E < = 10$ ft

Where:

G_p = minimum pedestrian crossing time, sec

L = Length of the crosswalk, ft

S_p = average walking speed of pedestrian, ft/s

N_{ped} = number of pedestrians crossing per phase in a single cross walk, peds

W_E = width of crosswalk, ft

2.3 Hardware-in-the-Loop Simulation

A Hardware-in-the-Loop-Simulation (HiLS) refers to a system in which some part of a pure simulation is replaced with an actual physical component. The concept of HiLS has been used in the aerospace industry for more than 40 years (Engelbrecht et al., 2001). In 1995, Urbanik and Venglar introduced the concept of HiLS in the field of transportation engineering for the “SMART” diamond project that included a real time traffic simulation model based on HiLS. In 1998, Bullock and Catarella described a real-time simulation environment for evaluating traffic signal systems, which uses the HiLS concept as a testing tool (Bullock and Catarella, 1998).

In the transportation engineering field, microscopic simulation is becoming an important tool to model real world traffic and transportation problems. The existing microscopic simulation engines or models use an internal software controller to perform the signal control operation during simulations. These internal controllers are able to simulate basic fixed time control operations, basic signal coordination and in some cases could simulate actuated controller logic. However, advanced signal control, advanced signal coordination, cyclic transition algorithms, signal preemption and transit priorities strategies are difficult to emulate using this internal software controller. Furthermore, the internal software controllers are not required to conform to any specifications or standards as required for the external traffic controller. Therefore, introducing an external actual controller can achieve high fidelity and increase credibility in the use of microscopic traffic simulation.

2.3.1 Hardware-in-the-Loop Simulation Framework

Figure 2.5 represents the schematic diagram of Hardware-in-the-Loop Simulation. Hardware-in-the-Loop Simulation consists of three main components: simulation engine, controller interface device (CID) and actual traffic controller. The external controller (hardware) is connected to the simulation engine through the interfacing device known as a controller interface device.

The CID communicates with the hardware component (actual traffic controller) and software component (microscopic simulation engine) of the system. The CID receives the phase indications from the actual traffic controller and passes on the detector call/actuation from the simulation engine to the external traffic controller. In other words, the input to the CID is the detector actuations from the microscopic simulation engine and the output from the CID are the phase indications from the external controller. In HiLS, the actual external controller replaces the internal software controller of the simulation engine.

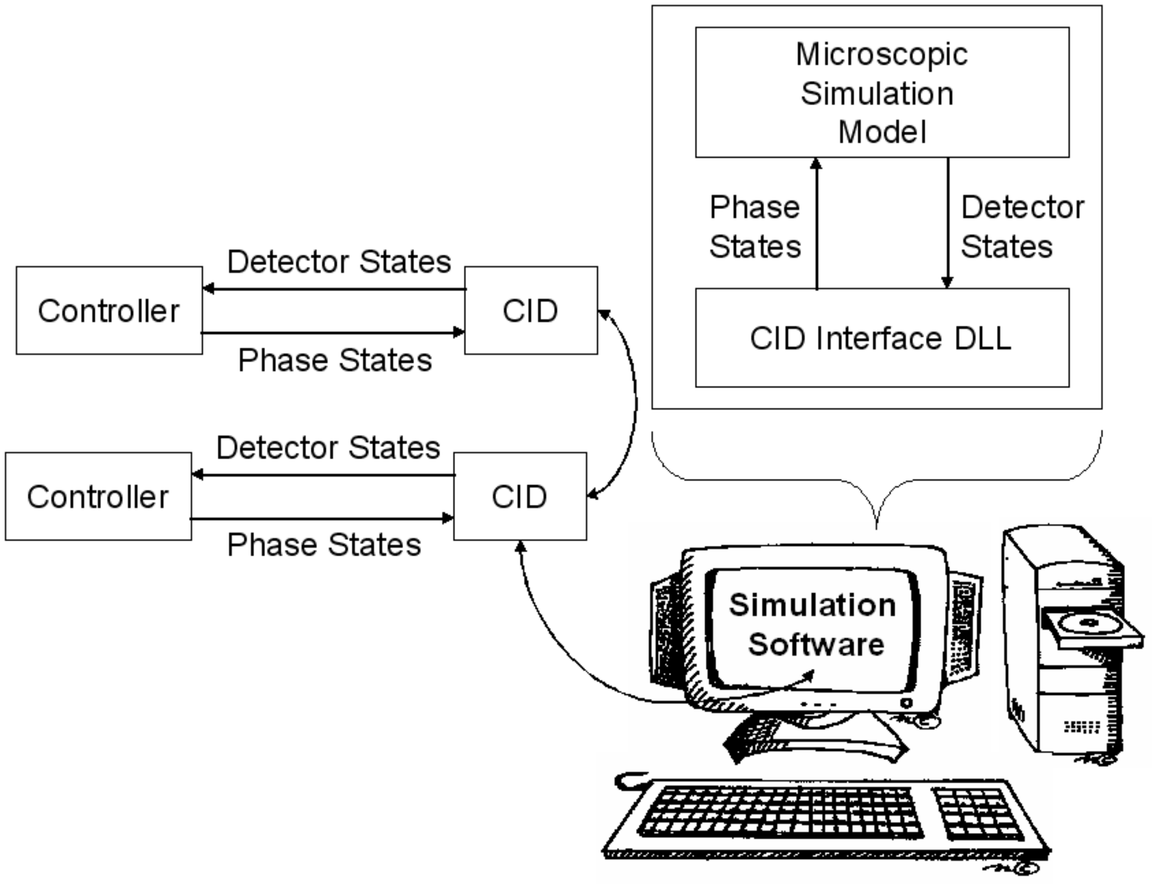


Figure 2.5: Schematic Diagram of Hardware-in-the-loop

2.3.2 Controller Interface Device (CID)

The CID is the critical component for Hardware-in-the Loop Simulation (HiLS). The CID is the interfacing device, which communicates with the hardware (external controller) and the software (simulation engine) component in HiLS. Through the CID the detector actuations from the simulation engine are sent to the external signal controller, and the phase indications from the external signal controller are sent back to the simulation engine. The connection between the CID and external controller depends on the type of controller. For the NEMA controller with TS 1 specification, the CID is connected with a standard TS 1 A, B and C connectors. For the 170 or 170E signal controller, which follows Caltrans (California Department of Transportation) specifications, the CID is connected with the standard C1 connector. In the case of a NEMA controller with TS 2 specification, the CID is connected through the Port 1 connector.

There are two different types of CIDs available to the research community. The first one, REL–CID was developed in 1995 by the Texas A&M ITS Research Center for Excellence (Engelbrecht et al., 2001). The Texas Transportation Institute, TransLink Research Center further developed this CID design, which was used in the "SMART" diamond project for its Roadside Equipment Laboratory (REL).

The second one, LSU-CID was developed at Louisiana State University for the Federal Highway Administration. Louisiana State University collaborated with ITT systems & Sciences Corporation (Formerly known as Kaman Science Corporation) to develop the LSU-CID. Engelbrecht et al (2001) compared the REL-CID and LSU-CID and summarized the findings as follows

- The REL-CID works on a parallel interface compared to the LSU-CID which works on a serial interface. The REL-CID transfers data at a higher rate and has a lower processor power requirement compared to the LSU-CID
- The LSU-CID could interface better with a notebook computer due to using a serial interface, thus making LSU-CID more portable
- The design of REL-CID is more flexible compared to the LSU-CID, so it could be reconfigured to connect to other control devices such as ramp meter controllers.
- The LSU-CID design allows multiple CID's linked in a daisy chain, where as in the REL-CID, due to the constraint in the I/O interface boards, a maximum of three controllers could be connected.

Though the LSU-CID and REL-CID were used for research purposes, they were not suitable for mass production. As part of the U.S. Department of Transportation's ITS initiative, the FHWA wanted the CID made available to practicing traffic engineers nationwide. National Institute for Advanced Transportation Technology (NIATT) designed and developed the CID-II for FHWA, which was based on the LSU-CID developed by Bullock (Bullock and Catarella, 1998). This project utilized the NIATT CID-II developed by University of Idaho, Moscow and manufactured by McCain Traffic Supply Inc.

2.3.3 Real-time Simulation Operation

HiLS is also known as real-time simulation, since the whole setup operates in real-time. The hardware (external traffic controller) can only perform in real time i.e., it operates in real clock time. Simulation models run in a simulation clock time, which is normally faster than the real clock time. To ensure the synchronization between the external controller and simulation model, the clock of the simulation model should match with the real time clock. The HiLS is feasible only when the simulation model can run faster than real time. In the case that the simulation model runs more slowly than real clock time (as might be the case in very detailed simulation of the physics of a system), HiLS is infeasible, as it would be virtually impossible to operate the traffic controller slower than the real clock time. Therefore, before setting up experiments using HiLS, it should be ensured that the simulation model can run fast enough to keep up with real clock time.. The guidelines proposed by Bullock et al (2004) to ensure real-time operation for the HiLS setup are as follows

1. The external controller and simulation engine should be synchronized to ensure that the simulation runs in real-time.
2. The simulation needs to run in evenly spaced simulation time steps.
3. The simulation of each time step must run faster than real-time, so that the interface software has time to run and wait for the real-time clock to reach the start of the next simulation period.

2.3.4 Advantages and Disadvantages of Hardware-in-the-Loop Simulation

Hardware-in-the-Loop Simulation is a new concept introduced to the field of transportation engineering. HiLS has proved to be a useful tool for traffic engineers in testing new control logic and utilizing advanced features of the controller. However, there are certain disadvantages using HiLS system. The advantages and disadvantages of using HiLS are listed as follows:

- **Advantages**

1. HiLS adds realism to the simulation as a real external traffic controller is used.
2. It increases the credibility of the results obtained from the microscopic simulation.
3. Instead of using the conventional suitcase tester to test the working of the signal controller HiLS can be used.
4. Capable of testing of advanced signal control strategies.

- **Disadvantages**

1. Traffic simulation models usually run faster than real time on smaller network and faster computers. Due to the use of HiLS, the simulation speed has to match with the real time clock, thus making the HiLS much slower than normal software simulation.
2. Computers that are more powerful may be required to perform HiLS experiments for large networks.
3. Special care should be taken to operate the HiLS in real time.

4. For testing multiple intersection signal controls, multiple pairs of CID and controller are required.

2.4 Implementation Framework

2.4.1. Overview

This chapter presents the implementation framework, which integrates the flow prediction module, signal optimization module and HiLS module in the proposed adaptive control system.

Figure 2.6 shows the entire system. The upper block represents the software component and lower block represents the hardware component of the system.

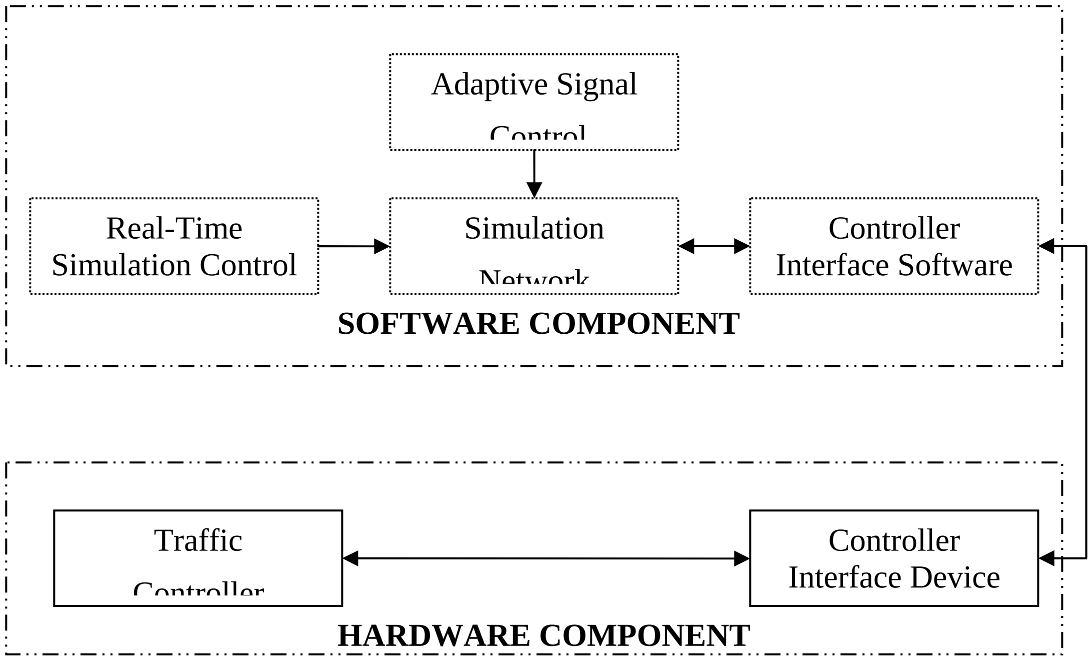


Figure 2.6: Overview of the Implementation Framework

As can be seen in the figure, the upper block consists of the adaptive signal control, real time simulation control, simulation network and controller interface software, while the lower consists of the controller interface device and an external traffic controller.

Adaptive signal control consists of the flow prediction and signal optimization module. The real-time control component operates the simulation to run in real time for the hardware in the loop simulation. Controller interface software, interfaces with the controller interface device to pass the detector actuations and to obtain the phase indications from the traffic controller.

Adaptive signal control, real-time control and controller interface software are plugged into the simulation program as plug-ins. Figure 2.7 shows the connectivity for the HiLS. The following sections of this chapter explain each component of the system in detail.

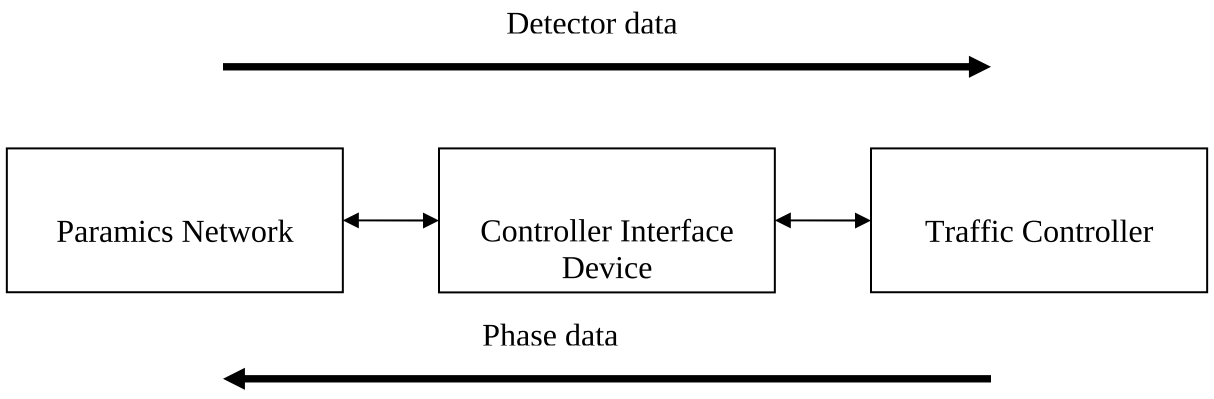


Figure 2.7: Hardware-in-the-Loop Simulation Connectivity

2.4.2. Software Components

Simulation Network

The network selected (see Figure 2.8) to test the proposed adaptive control system consists of five intersections, which are located in downtown Logan, Utah. Here the proposed adaptive signal control logic is implemented on one of the five intersections. The intersection under consideration is on 400N and US-91 (primary signal). The primary signal is surrounded by four satellite signals namely 200N&US91, 400N&100W, 500N&US91 and 400N&100E on the south, west, north and east directions respectively.

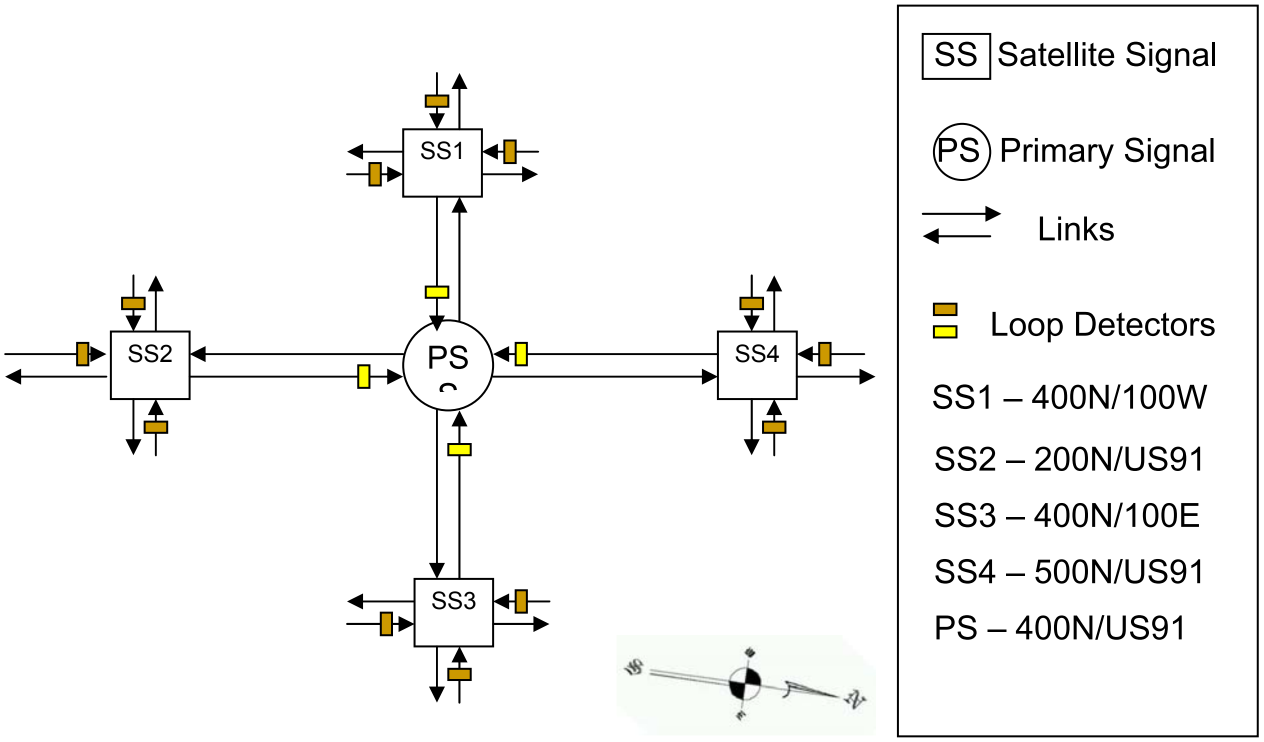


Figure 2.8: Schematic Diagram of the Proposed Network

To provide a desirable environment to the prediction model, it is assumed that no interlink sources or sinks are in between the links connecting the intersections. Loop detectors are installed on all of the approach links of the satellite intersections. These detectors are located at 125 ft upstream to the signals. To assess the quality of the traffic flow prediction for the primary signal, detectors are installed on all the approaching links at 200 ft upstream to the primary signal.

Paramics (Parallel Microscopic Simulation)

Paramics is a well-known, state-of-the-art microscopic traffic simulation software (Duncan, 1995). It is a scalable, user programmable and computationally efficient traffic simulation modeling software. It has the capabilities to be customized using Application Programming Interface (API). Using the API functionality, the user could modify the underlying simulation model and add complementary modules to the simulation engine. The API functions in it can be divided into three categories namely, overload, override and callback functions (Lee et al., 2001). Using the overload function, one could attach additional routines to the inbuilt functions. To replace the existing inbuilt functions in Paramics, Override functions are used. The call back function helps the user in gathering the data, for the calculation of MOE's (measure of effectiveness) required from the simulation. Some of the applications that could be tested using the API functionalities are signal optimization, adaptive ramp metering, incident management and so on. In this study, the adaptive control logic is built using the Paramics API functionalities and tested using the Hardware-in-the-Loop Simulation.

Adaptive Signal Control

This component consists of the flow prediction and the signal optimization models. *Adaptive.dll* is the dynamic link library that constitutes these two models. The description of these models is given by the flow chart in Figure 2.9.

This flowchart explains the development framework for the event-based short-term flow prediction model used in this project. The function *qpx_NET_postOpen* initializes and obtains the information regarding the network geometry. The functions *open_file* and *open_adaptive_control* are the subfunctions under *qpx_NET_postOpen*. The function *qpx_VHC_releas*e associates a user defined data structure for each vehicle released from the zones. When a vehicle passes any of the four advanced satellite detectors of the upstream intersections, the function *qpx_VHC_detector* calls the *adaptive* function, which is the core function in *adaptive.dll*. The sub functions of the signal control function (*adaptive),* checks the phase status of the corresponding signal. The *Adaptive* function calculates the signal delay and adds the queue delay if any vehicle queue exits. Travel times from the advanced detectors to the stop bar and from the stop bar to the subject intersection are then calculated. Using these delays the arrival times of the vehicle to the primary intersection is finally predicted.

This predicted arrival time is sequentially stored in a database at pre-specified prediction frequency (in our case 5 seconds). The database allows us to calculate the traffic flows at pre-specified time horizons. Thus, from the information gathered as above, one could infer on the time distribution of vehicle arrivals at the primary signal. However, our proposed adaptive signal just uses a cycle-by-cycle prediction where in the flow that is predicted in the current cycle would determine the traffic parameters settings for the next cycle.

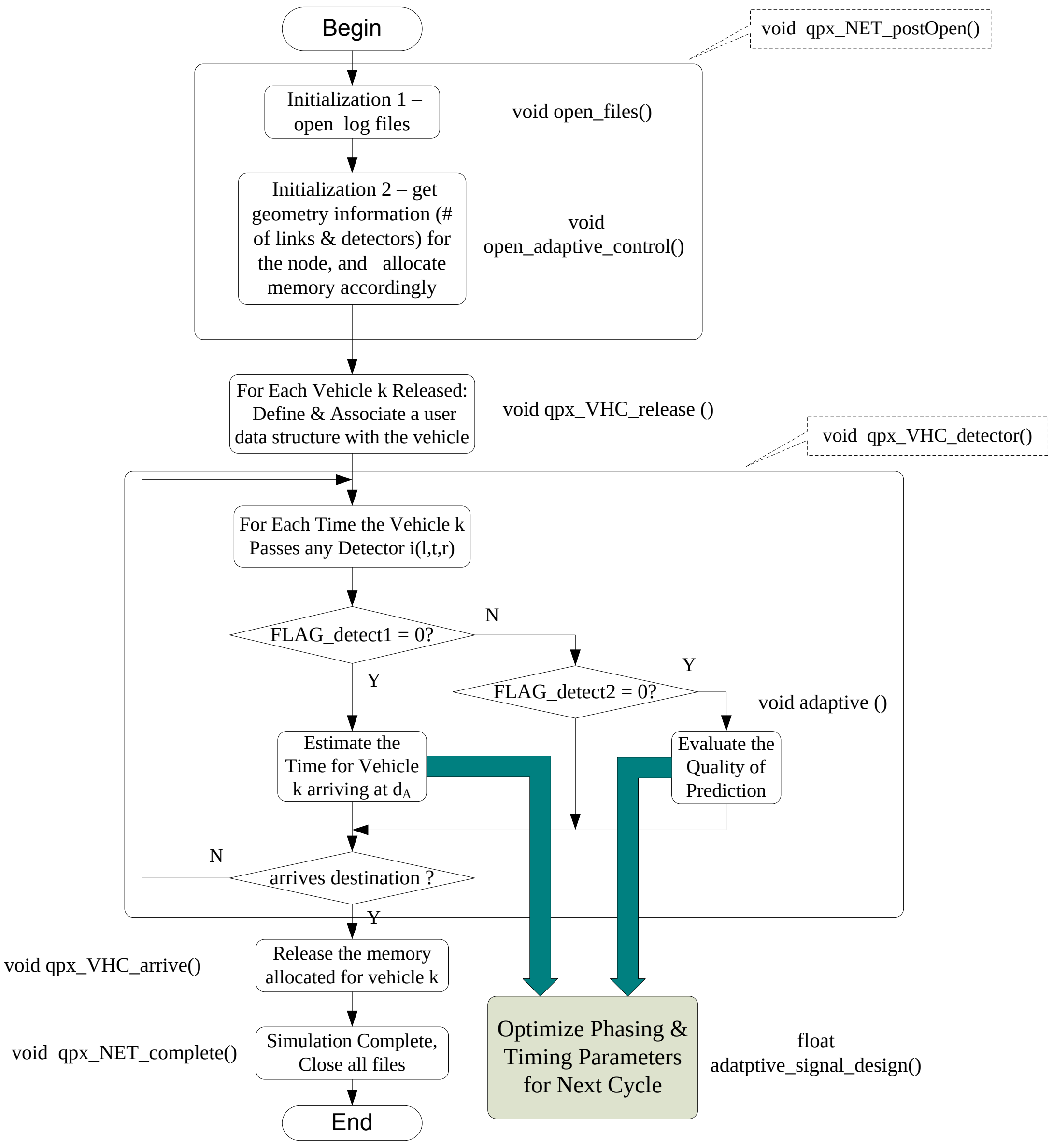

Begin
void qpx_NET_postOpen()
Initialization 1 – open log files
void open_files()
Initialization 2 – get geometry information (# of links & detectors) for the node, and allocate memory accordingly
void open_adaptive_control()
For Each Vehicle k Released: Define & Associate a user data structure with the vehicle
void qpx_VHC_release ()
void qpx_VHC_detector()
For Each Time the Vehicle k Passes any Detector i(l,t,r)
FLAG_detect1 = 0?
N
Y
FLAG_detect2 = 0?
Y
void adaptive ()
Estimate the Time for Vehicle k arriving at d_A
Evaluate the Quality of Prediction
N
arrives destination ?
Y
void qpx_VHC_arrive()
Release the memory allocated for vehicle k
void qpx_NET_complete()
Simulation Complete, Close all files
Optimize Phasing & Timing Parameters for Next Cycle
float adatptive_signal_design()
End

Figure 2.9: Flowchart for Adaptive Signal Control

Finally, a traffic signal timing plan is designed from predicted flows obtained in the previous cycle. The function *adaptive_signal_design* designs the signal timing plan for the subsequent cycle. We split the demand for permitted and protected phasing as follows.

1. If the Left turn volume is greater than 240vphpl or the opposing through lane volume times the Left turn volume is greater than 50000vphpl, then a protected + permitted phasing is designed. Else, the left turn is designed as a permitted phasing.
2. For a protected + permitted phasing, if the left turn volume is less than 240vphpl, then design only for the protected phase. The permitted phase is omitted in this case
3. For a protected + permitted phasing, if the left turn volume is greater than 240vphpl, then design 240vphpl as a protected phasing and the remaining volume as a permitted phase.

These steps explain the design procedure for a protected + permitted left turn phase, which is not explained in HCM .

Real-time Simulation Control

Real-time control synchronizes the operation of the simulation engine and the traffic controller for the Hardware-in-the-Loop Simulation. The dynamic link library *Realtime.dll* controls the simulation program in real time. The Figure 2.10 systematically explains the operation of the real-time interaction between the real clock time and the simulation clock time.

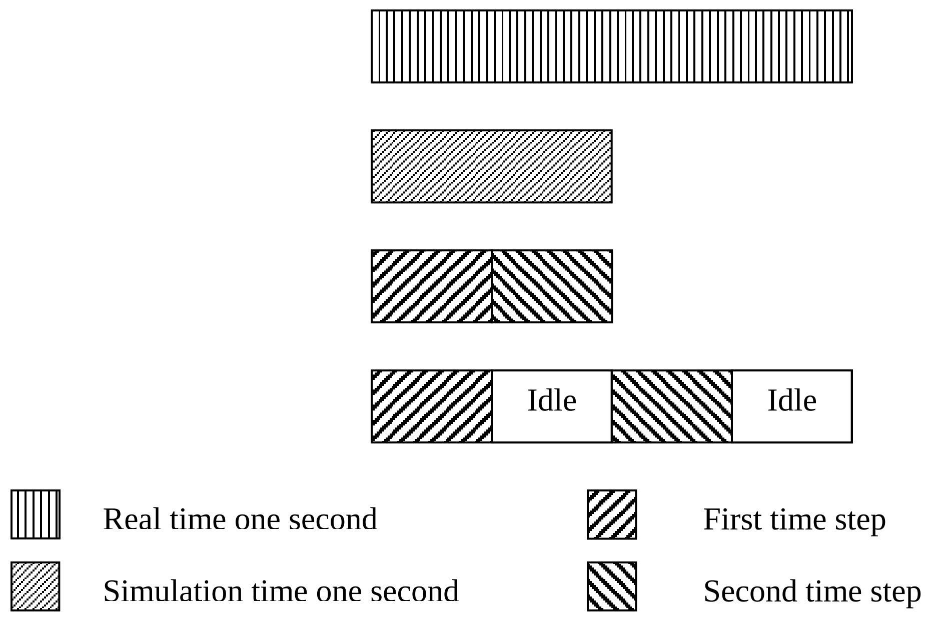


Figure 2.10: Real-time Control Operation

In Paramics, a minimum of two time steps constitute a simulation second. But for simulation second to match with the real clock second, each time-step should complete the execution in exactly 500 real clock microsecond. Therefore, after each time-step the simulation is made to wait for the remaining clock time until it reaches 500 real clock-time microseconds. The real-time control process consists of the following steps:

1. Simulation is set to run in the single step mode.
2. Specify the number of simulation time-steps to be executed in a simulation second.
3. Record the start time (real clock) at the beginning of the time step, ST.
4. Record the end time (real clock) at the ending of the time step, ET.
5. Calculate the lapse time (lapse time = 500 – [ET – ST]).
6. Idle the simulation for lapse time.

7. Start the next time-step.
8. Go to step 3.

Controller Interface Software

A software interface, interface.dll, provides the linkage between the CID and the microscopic simulation software Paramics. Interface.dll is a dynamic link library (DLL) software module, which operates in a Windows operating system. The transfer of the data between the PC and the CID is through a Universal Serial Bus (USB).

The CID API version 1.0 provides ten function calls. These API functions could be used by the programmer to directly communicate with the CID hardware. The programmer can also use these API functions to interface the NIATT CID with their own simulation software (example Corsim, Paramics, Vissim) which are capable of using API. Some of the important API function call used to communicate with the CID hardware is listed below.

1. int GetOnlineCIDs(int cidIDBuffer[]):

   Input: cidIDBuffer[], integer array for all CID ID

   Return: (1).The number of the online CIDs

   (2).Integer array for all CID ID's (cidIDBuffer[])

This function opens all the CID's on the USB bus and saves all the CID ID's to an integer buffer. The USB handle is initialized for each CID device found on the USB bus. The CID ID's are placed in the integer array.

2. int SetChannelStatus(int cidID, int channellID, int timing)

   Input :(1). CID ID number

   (2). Channel ID number

   (3).Timing – The channel active timing

   Return: (1). True – success

   (2). False – failure

Using the above function, the CID channel status and timing are set. The number "0" is used to set the channel active forever, and other numbers are used to set some duration of timing. These are in milliseconds.

3. int WriteCIDEx(int cidID)

   Input: CID ID number

   Return: (1) The number of bytes actually sent to the controller

   (2) -1 (failure)

This function writes the data into the CID. The channel status should be set using the SetChannelStatus function before writing it using the function WriteCIDEx.

4. int ReadCIDEx(int cidID)

   Input: CID ID number

   Return: (1). The number of byte actually read from the controller

   (2). -1 (failure)

This function saves the CID output status in an internal buffer. To get the specific phase status of the phase from the internal buffer, the function GetPhase is called.

5. int GetPhase(int cidID, int phase)

   Input: (1)CID ID number

   (2) phase number

   Return: (1) 1 – Red light active

   (2) 2 – Yellow light active

   (3) 4 – Green light active

   (4) -1 (failure)

This function reads the specific phase status from the CID internal buffer. Before using this function, the ReadCIDEx function is called to update the internal buffer that is having the current traffic controller phase status.

2.4.3. Hardware Components

Controller Interface Device

The implementation of the Hardware-in-the-Loop Simulation uses a microscopic simulation program (Paramics) with traffic signal control hardware. An interface device, known as the Controller interface device (CID) is used to provide a real-time link between the simulation program and the traffic controller. This project utilizes the CID II developed by the National Institute for Advanced Transportation Technology (NIATT). The NIATT's CID II is capable of providing a real-time interface between a 170, 2070 and NEMA TS1 and TS2 traffic controller (Kyte et al., 2001). Figure 2.11 shows the NIATT/Mc Cain CID - II

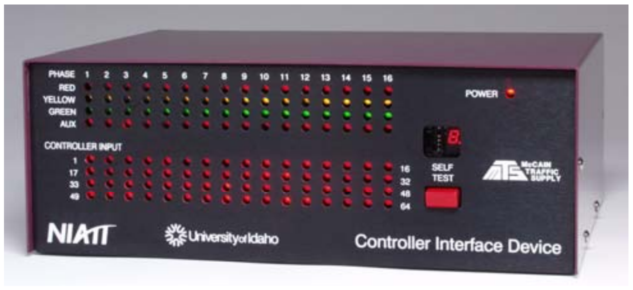


Figure 2.11: NIATT/Mc Cain CID - II

Traffic Controller

The traffic controller used in this project is a NEMA TS 1 specification controller. The data transfer between the NEMA controller and the CID is through the connectors A, B, C of the NEMA controller. Figure 2.12 shows the connectivity between Paramics, CID II and NEMA controller.

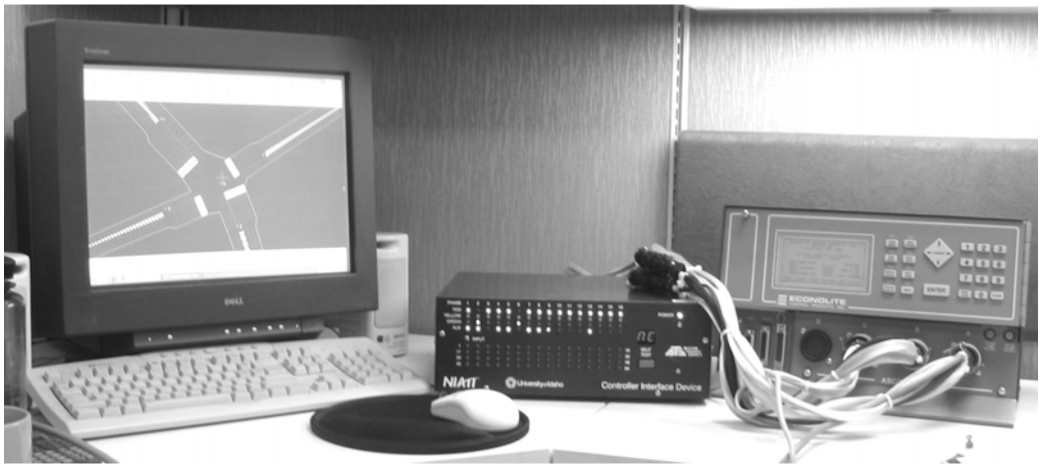


Figure 2.12: Connectivity between CID-II, Paramics and NEMA Controller

2.5 Simulation Experiments and Results

This chapter compares the performance of the proposed adaptive control system via simulation experiments. The proposed system is tested and evaluated using the Hardware-in-the-Loop Simulation. The first section of this chapter compares the HiLS and standard microscopic simulation (SMS) for pre-timed and actuated signal control.

The microscopic simulation program that is used in the HiLS is Paramics. The controller interface device (CID), which interfaces between the microscopic simulation and the traffic controller, is the NIATT/Mc Cain CID – II. The traffic controller used in these experiments is the ASC/21-2100, a NEMA TS 1 specification controller, manufacture by Econolite control products, INC.

2.5.1. Comparison of HiLS and SMS

Comparison of the HiLS and standard microscopic simulation were performed to study the effect of the traffic controller on the flow of traffic and the performance of the signalized intersection. The intersection selected for the experiment was Alton and Irvine Center Drive, Irvine California. This intersection is operating under an eight-phase fully actuated signal control. The base signal timing plan for the pre-timed and actuated control is shown in Table 2.13.

Table 2.13: Signal Timing Plan for Alton and Irvine Center Drive

	Phase (sec)							
	ϕ-1	ϕ-2	ϕ-3	ϕ-4	ϕ-5	ϕ-6	ϕ-7	ϕ-8
MIN Green	5	10	5	8	5	10	5	8
MAX Green	24	32	24	32	24	32	24	32
Vehicle Extension : 2 sec, Cycle length = 120 sec								

The performance of hardware and software controllers are compared for pre-timed and actuated signal control logics. Table 2.14 tabulates the link flows for the SMS and HiLS operation. While the software and hardware controller showed a similar performance for pre-timed signal logic (1-2 % difference), the latter showed a higher difference of four to five percents in link flows.

Table 2.14: Comparison between SMS and HiLS for Link Flows

	Pretimed Control				Actuated Control			
Link Flow	HiLS	SMS	Difference	% Difference	HiLS	SMS	Difference	% Difference
North Bound, vph	838.4	863.2	24.8	2.8	829.6	803.2	26.4	3.3
South Bound, vph	748	740.8	7.2	1.0	759.2	709.6	49.6	7.0
East Bound, vph	899.2	897.6	1.6	0.2	910.4	909.6	0.8	0.1
West Bound, vph	847.2	880.8	33.6	3.8	877.6	946.4	68.8	7.3
Average difference				1.95	Average difference			4.4

Table 2.15 compares the control delay and LOS for SMS and HiLS. The control delay and LOS measure the performance of the intersection. As it can be seen from the table for both controller types (pre-timed and actuated), operating on HiLS and SMS, the difference between control delays is negligible. The LOS for pre-timed control is ‘C’ and for actuated control is ‘B’ for HiLS and SMS.

Table 2.15: Control delay and LOS

	Pre-timed		Actuated	
	HiLS	SMS	HiLS	SMS
Control Delay	38	37	20	19
LOS	C	C	B	B

2.5.2. Simulation Scenarios

Simulation experiments were conducted for evaluating the performance of the proposed adaptive control system. The performance of the proposed system is compared with the actuated control system with different demand levels (75%, 100%, 125% and 160%). A combination of eight scenarios were studied and compared.

For the simulation runs, Paramics is run from 7:30 to 8:30 AM, corresponding to the morning peak hour preceded by a fifteen-minute “warm-up” period in which the network is loaded. Warm-up period of fifteen minute is sufficient for the network to reach equilibrium, as the vehicle travel time is well below this value. The simulation is run with a fixed demand level for the entire period.

Traffic Condition and Signal Timing

The study site for the experiment is the intersection 400N and US-91. Presently this intersection works on actuated control logic. For the purpose of the experiment, the turning movement traffic counts (AM peak) from the UDOT’s Synchro model were obtained. No information was given on the days that the traffic counts were conducted. Table 2.16 provides the summary of the turning movements and Figure 2.17 represents the intersection turning movements displayed in the Synchro model.

Table 2.16: Summary of turning movements

Intersection	NB			SB			EB			WB		
	R	T	L	R	T	L	R	T	L	R	T	L
400 N./Main	346	750	23	22	570	76	25	200	39	64	132	158
200 N./Main	52	972	103	99	598	56	39	108	125	22	115	23
500 N./Main	26	801	26	58	597	47	47	49	19	30	65	24
400 N./100 W	346	750	23	11	137	20	25	200	39	38	94	74
400N./100 E	52	972	103	20	611	31	39	108	125	31	308	19

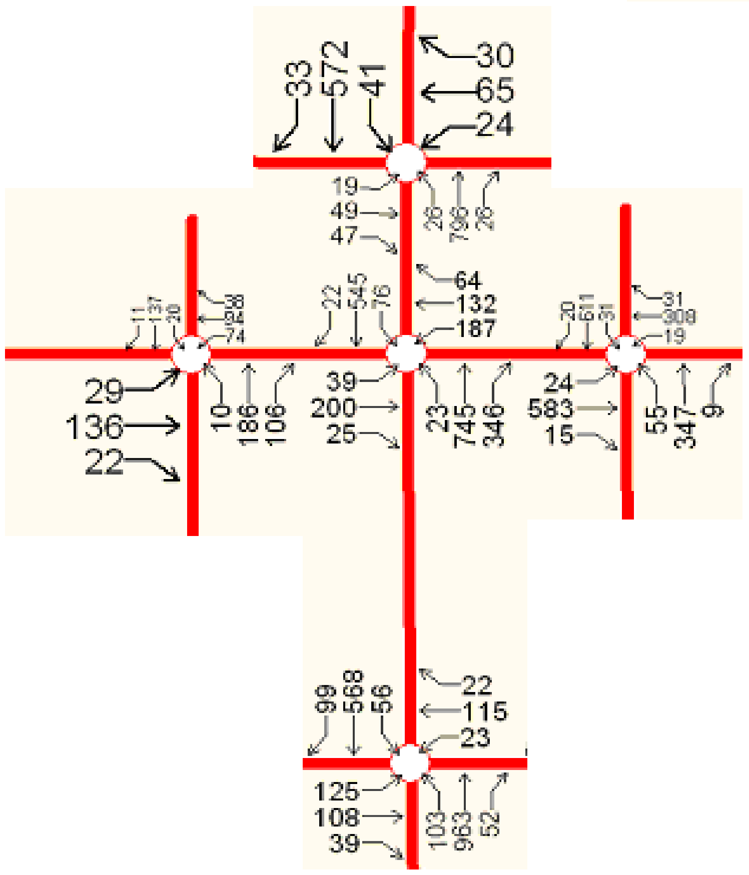


Figure 2.17: Network Modeled in SYNCHRO

In addition to the turning movement traffic counts, the existing signal timing plan was obtained from the UDOT's Synchro model. The base signal timing plan for all the intersections were generated via Synchro off-line signal timing optimization program. For more information on the existing signal timings, refer to Table 2.18: Summary of Existing Signal Timing Plans for Logan Main Street Network.

Table 2.18: Signal Timing Plan Details

Intersection	**Phase (sec)**								**Cycle Length**
	ϕ-1	ϕ-2	ϕ-3	ϕ-4	ϕ-5	ϕ-6	ϕ-7	ϕ-8	
400 N./US-91	11	26	11	22	11	26	11	22	70
200 N./ US-91	68	12	30	--	68	12	30	--	110
500 N./ US-91	80	30	--	--	80	30	--	--	110
400 N./100 W	21	34	--	--	21	34	--	--	55
400N./100 E	29	26	--	--	29	26	--	--	55

OD Demand Generation

The origin-destination demand of the simulation model was generated using the Paramics Estimator. The Estimator is an OD matrix estimation tool, which is designed to integrate with the Paramics module. The estimator utilizes all the aspects of the core Paramics simulation, ensuring maximum compatibility between the OD matrix, model structure and the underlying assignment technique. The GEH value obtained from the estimator for the calibration of the model is 0.5 which is lower than five, thus satisfying calibration acceptance criteria.

Determination of Number of Simulation Runs

This section presents the determination of the number of simulation runs required to obtain a stable simulated result on which the analysis and credible conclusion can be made. Paramics is a stochastic simulation model, which rely upon random number to release vehicles, assign vehicle type, select their destination and their route, and to determine their behavior as the vehicle move through the network. Therefore, multiple simulation runs using different seed numbers are required and the median simulation run (based on a user-specified measure) or the average results of several simulation runs can reflect the average traffic condition of a specific scenario(Chu et al., 2004). Equation 5.1 can be used to determine the sufficiency of the number of simulation runs.

$$N = \left(t_{n-1,\alpha/2} \cdot \frac{\delta}{\mu \cdot \varepsilon} \right)^2 \qquad \textbf{(5.1)}$$

where: N is the number of simulation required, tn-1,α/2 is the t-statistic with n-1 degrees of freedom and (100-α)% confidence level, n is the number of simulation runs currently performed, μ and δ are mean and standard deviation of the current simulation result, and ε is the allowable error specified as the fraction of current mean (e.g., 10% of μ).

In this study, the control delay for the primary intersection was considered as the performance measure. Using equation 5.1, the number of runs was calculated. For a confidence interval of 90%, 4 to 6 simulation runs were required depending on the scenario. However, due to the small number of simulation runs required, nine simulation runs were performed for all the scenarios in the study.

2.5.3. Results

In this section, the results of the performance of adaptive control system against pre-timed control and actuated signal control under different performance measures are shown.

Travel times predicted by the PREDICT model correlated well with the observed travels times. The error in the prediction was in the acceptable range of 10-12% and showed a normal distribution (Figure 2.19). The normal distribution of the error corresponds to white noise rather than a systematic error.

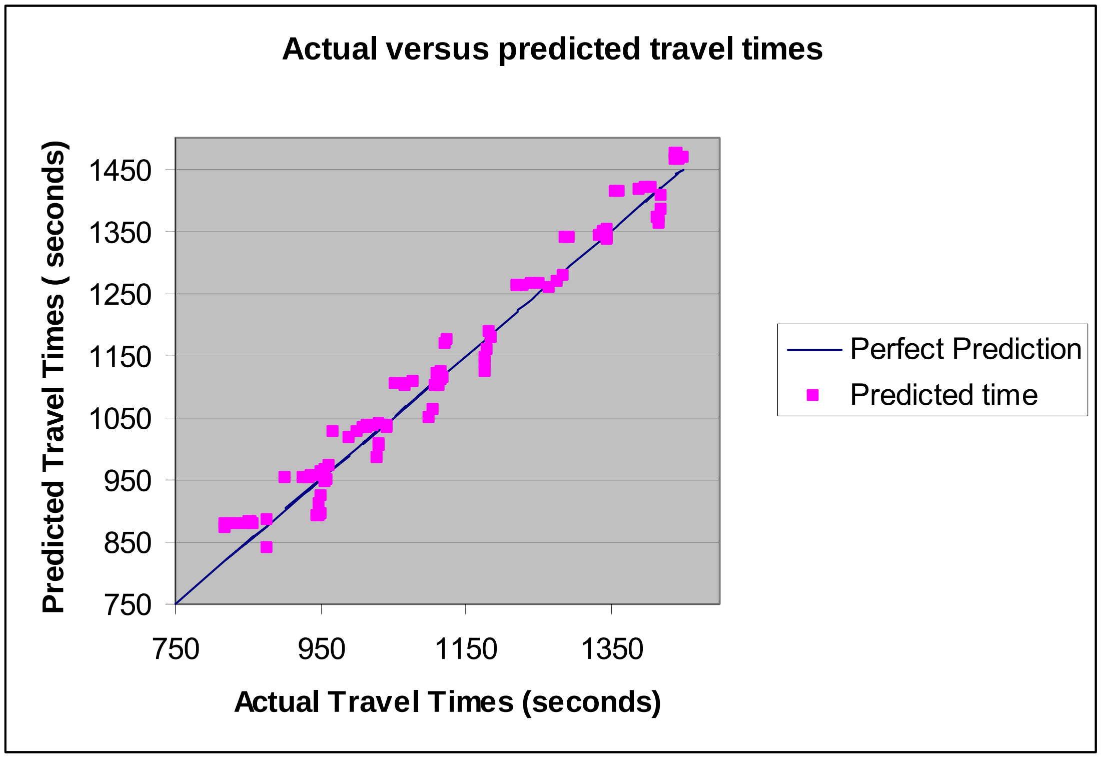


Figure 2.19: Actual verses Predicted Travel Times

Figure 2.20 shows the actual and predicted flow profiles. The x axis represents the time series which is divided into 5 second time intervals. The y-axis represents the number of vehicles. From the plot, it can be inferred that the actual flow profile matches closely with the predicted flow profiles. Only a portion of the simulation time for the actual and predicted flows is shown in the figure.

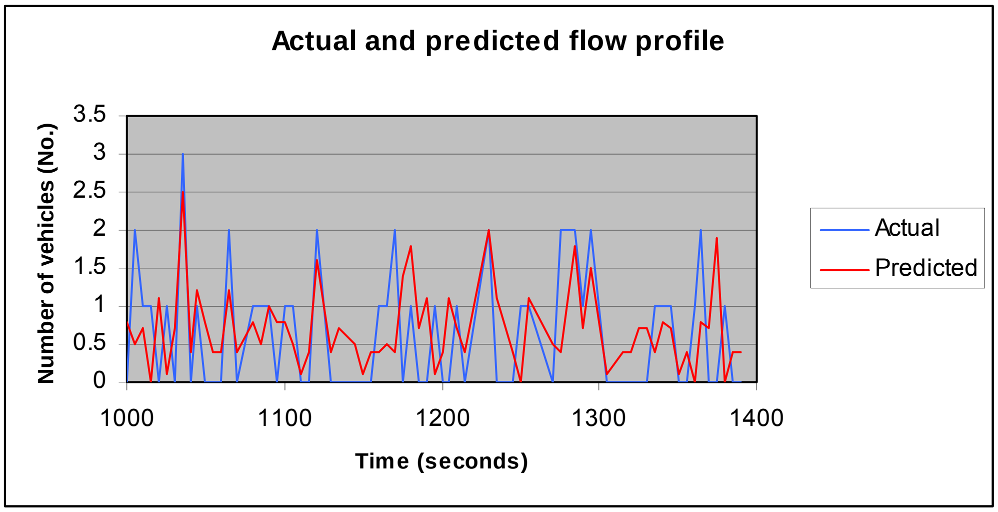


Figure 2.20: Flow Profile Showing Actual and Predicted Number of Arrivals

The choice of an optimal control device is necessary for a sustainable signal control in lieu of increasing future demands. One way of selecting an optimal control device is through intense scrutiny of control delays. In the present study, it was found that adaptive signal control performed better than actuated control, especially under conditions of high demands and various kinds of special-event activities (125 – 160%; Table 2.21). Under low traffic demand conditions, both the signal control systems display a similar performance. While the LOS for actuated controls dipped down to a grade of “E” under high demand conditions, the same for adaptive signal control depreciated by one level to “C”.

Table 2.21: Control Delay and LOS

	75 %		100%		125%		160%	
Control Type	ADP	ACT	ADP	ACT	ADT	ACT	ADP	ACT
North Bound, sec	8	4	9	7	12	27	25	124
South Bound, sec	16	14	15	19	19	21	26	38
East Bound, sec	25	16	28	20	34	25	40	26
West Bound, sec	24	13	29	15	31	31	45	82
Control Delay, sec	18	12	20	15	24	30	34	67
LOS	B	B	B	B	C	C	C	E
Adaptive signal control = ADP, Actuated signal control = ACT								

The better performance shown by the adaptive signal control is further clarified in the estimates of intersection through put and link delay (Table 2.22 and Table 2.23 respectively).

Table 2.22: Link Queues for 400N and US-91

	Adaptive signal control				Actuated signal control			
Demand, %	75	100	125	160	75	100	125	160
North Bound, sec	3	4	3	4	2	3	4	10
South Bound, sec	0	3	3	3	0	2	2	2
East Bound, sec	3	2	5	5	0	2	4	5
West Bound, sec	2	4	4	4	2	3	3	9

Table 2.23: Intersection Throughput

	75 %		100%		125%		160%	
Control Type	ADP	ACT	ADP	ACT	ADT	ACT	ADP	ACT
North Bound, no.veh	810	817	1113	1077	1441	1369	1818	1574
South Bound, no.veh	470	489	610	674	781	788	1049	1018
East Bound, no.veh	199	222	248	255	327	319	410	459
West Bound, no.veh	296	285	375	374	497	454	644	592
Intersection Throughput	1776	1813	2347	2380	3046	2930	3922	3643
Adaptive signal control = ADP, Actuated signal control = ACT								

Table 2.24 gives the estimates for signal timing for different phases under varying conditions of demand. It is clear that phase 2 and phase 6 have almost double the signal time that the other phases as they represent the north south bound through traffic. The cycle length changes only from 61 seconds to 91 seconds when the demand is increased from 75% to 160%. This clearly points to a high volume capacity of the intersection and ability of the intersection to serve greater demands.

Table 2.24: Average Signal Timing Plan Details for Adaptive Signal Control

Demand	**Phase (sec)**								**Cycle Length**
	ϕ-1	ϕ-2	ϕ-3	ϕ-4	ϕ-5	ϕ-6	ϕ-7	ϕ-8	
75 %	11	28	12	15	11	28	11	15	66
100%	11	33	14	16	11	33	11	16	74
125%	11	39	16	18	11	39	11	18	84
160%	11	43	19	18	11	43	11	18	91

2.6 Event-based Short-term Traffic Flow Prediction

2.6.1. Introduction

Traffic flow prediction is one of the most important components for the development of adaptive traffic signal control strategies. The effectiveness of the traffic control system highly depends on the accuracy of the predicted traffic flow information, i.e. the traffic pattern along the space and time horizons.

Adaptive signal control systems are advanced traffic control strategies that can generate optimal signal plans according to online traffic data from the surveillance systems. In contrast to some earlier responsive traffic control systems, which also use real-time traffic measures but matching traffic conditions with existing timing plans, most adaptive control systems recognize the importance of traffic flow prediction and use online computers to create optimal signal plans. Adaptive signal control systems can not only adjust signal timing during different time intervals of a day, but also accommodate for non-recurring events, such as accidents which may happen in the network.

Adaptive signal control systems need accurate and real-time traffic flow information to efficiently optimize timing plans at signalized intersections. Accurate data with acceptable error and less lagging are important for the traffic measurements. Comparing with the previous systems that are based on historical data, real-time is the advantage of adaptive control systems. Without timely traffic information, system will lose responsive ability to traffic flow fluctuation in the network. On the other hand, accurate traffic data is important to implement the traffic control system; because all generated optimal timing plans are calculated based on measured traffic information. If the field data sent to the central computers are not correct, the corresponding created timing plan will not adapt to real traffic situation, and then the objectives of minimize vehicle delay and maximize throughput will not be achieved.

Adaptive signal control system must have ability to predict the current traffic conditions at least for a short time. However, the network surveillance systems that deployed by transportation agencies can not prior generate timely traffic flow information without appropriate prediction model. The surveillance infrastructures, including loop detectors, radars, and video cameras, can only measure, while not predict real-time traffic flow. Therefore, the system can only reactively extend or terminate current running phase, and do not have the ability to update the phase sequence or generate new timing plans for the next cycles. The complicated arterial networks need adaptive signal control system that can proactively optimize signal plans based on predicted real-time traffic conditions.

2.6.2. Literature Review

The development of adaptive control systems was studied since the early of 1970s, when the Federal Highway Administration (FHWA) started to develop UTCS to test variety advanced traffic control strategies (Tarnoff, 1975 and Gartner, et. al 1995). Time lagging is the disadvantage of UTCS and the prediction model of UTSC can not generate accurate results if the surveillance detectors are outage (Kreer, 1975). SCATS and SCOOT are enhanced adaptive control system over the first generation of UTCS that are widely implemented in the world (Sims, 1979, Lowrie, 1992, Robertson and Bretherton, 1991). SCATS can adjust traffic signal timing splits to accommodate traffic conditions based on minute by minute changes in traffic flow at each intersection. The detectors used for collecting traffic conditions are located at the stop line. The adjustments of signal timings in SCATS are incremental as opposed to global optimization. SCATS measures, while not predicts, traffic flow file in the network. Therefore it is only a reactive control system, and do not operate well when the network is congested. SCOOT uses system detectors, which located at the upstream end of the approach link, along with the predetermined travel time and degree of saturation to predict queues. Similar with SCATS, it makes incremental adjustments to the current signal plan for the next cycle, in response to changing traffic demands. Therefore SCOOT is also not a proactive control logic that would operate fairly well when the traffic condition in the network is unstable.

Some other adaptive control strategies, such as OPAC and RTACL allow proactive responses, and produce improvement on traffic delay reduction at signalized intersections. The major drawback of their prediction models is they do not fully utilize the available traffic information to estimate traffic flow pattern. Most of these logics only take into account the data from surveillance system, while the signal status data are omitted or less considered. Signal status information is especially important for the traffic flow prediction model when the signal timing plans implemented are not fixed-time. Head proposed a traffic flow arrival estimation algorithm — PREDICT to improve the effectiveness of calculating online phase timings (Head, 1995). In the PREDICT algorithm, the detector data of approaches at every upstream intersection, together with the traffic state (arrival and queues) data, and the control plans of the upstream signals are used to predict future traffic volume.

A major drawback of previous research is the high implementation cost in real world. Some models need additional investment for extra hardware to be installed to collect special data, which is prohibitively expensive especially on large arterial networks. Therefore, there is a need to develop prediction logics with the existing infrastructures, such as the installed inductive loop detectors and the on-site signal controllers.

This research proposed an appropriate traffic control model with forecasting ability to proactively optimize signal timings with the existing surveillance data, the signal status data, and the relationship between them. We try to fully utilize all available traffic information to better predict the traffic pattern in the network, and then the control system can generate better optimization timing plans based on the accurate traffic flow prediction. The logic proposed will not only fit stable for un-congested traffic network but also accommodate with fluctuated and non-recurring congested network.

2.7 Green Phase and Red Phase Actuation Model

2.7.1. Background

Traffic flow prediction is to predict the traffic flow pattern before the vehicles arrive at the subject intersections. Surveillance traffic data, such as vehicles' passing time at upstream detectors, are used to estimate vehicle arrivals. Signal status data, including corresponding signal running phases and signal phase states, are important for the prediction model to accurately calculate vehicle arrival time. Moreover, the geometric information, such as vehicles' traveling distance, is also needed.

The predict model needs to estimate not only the time when vehicles will arrive at the intersection, but also the number of arrival vehicles during the studied time interval. Vehicle counts can be measured from surveillance systems; however, most existing surveillance detectors can only measure the total number of passed vehicles, but cannot distinguish their moving directions. Therefore the number of vehicles that pass the upstream intersections to the subject intersection cannot be directly measured from the detectors. The most common method to solve this problem is to estimate the turning proportions at the upstream intersections. The estimation of arrival flow at the subject intersection is equal to the estimation of the turning proportions at the upstream intersections.

This research proposes a dynamic data-driven short-term traffic flow prediction model for signalized intersections. The model includes two sub-models: one is the vehicle arrival time prediction model and the other is the turning proportion prediction model. The dynamic vehicle arrival time and arrival flow rate that predicted by this model can thus be transmitted to the adaptive control system to generate optimal signal timing plans.

Figure 2.25 depicts the intersection geometry of the prediction model. Suppose intersection *A* is the subject intersection the traffic pattern of its arrival flow need to be predicted. For simplification, we only estimate the flow of the westbound, and the flows of other approaches can be calculated with similar procedures.

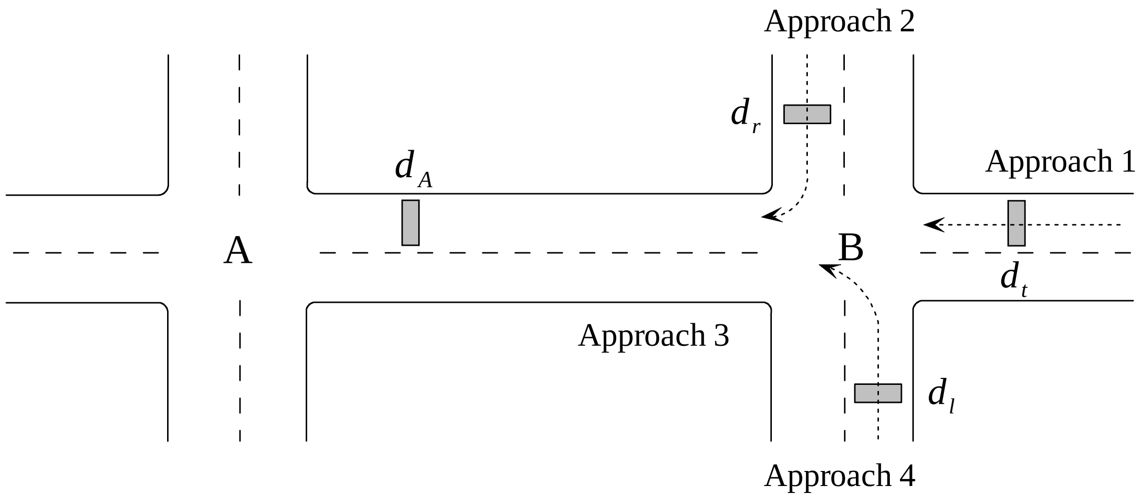


Figure 2.25: Geometric Layout of Prediction Scenario

As shown in Figure 7.1, detector d_A is an advanced detector that deployed at the westbound of intersection *A*, intersection *B* is its corresponding upstream intersection. Detectors d_l, d_r, d_t are advanced detectors installed at the approaches of intersection *B*. The detectors d_l, d_r, d_t can measure the flows that will turn left, turn right and pass through respectively. It assumes each vehicle actuation on any upstream detector d_i, i.e. d_l, d_t, or d_r, is an event, and the events on different advanced detectors are independent. When an actuation event happens, i.e. a vehicle passes an upstream detector d_i, all available traffic data at intersection *B* will be collected and transmitted to the central computer to predict the arrive time of the vehicle at detector d_A. The data include the vehicle's passing time, the corresponding signal status, such as running phase, phase timing parameters, at intersection *B*, and the distance from intersection *B* to detector d_A. Meanwhile, the data are also used to estimate the turning proportion of the flow at detector d_i, and then the traffic flow rate at detector d_A can be predicted. Therefore, the flow pattern at intersection *A* can be predicted when the vehicle is still running at the upstream links, and the system can proactively update the phase timing of intersection *A* to optimize traffic control.

The eight-phase dual-ring actuated signal concurrent controller is assumed to be adopted by intersection *B*. In fully-actuated signal control, all phases at an intersection are actuated. Therefore the length of each phase, and consequently the cycle length, will vary at each cycle. Some phases may be skipped if there is no vehicle actuation. The detailed description on how actuated signal works can be found in the textbook by McShane et al. (2004).

2.7.2. Green Phase Actuation Models

Assume the yellow and all-red intervals can be counted as green time, and then any event at detector d_i always occurs at two alternative states: actuation at green phase or actuation at red phase, i.e. the vehicle will arrive at the detector d_i either at green phase or at red phase. Therefore, the predict model can be divided into two parts according to the corresponding states: the green phase actuation model and the red phase actuation model. The details of the two models are discussed as follows in the section.

The green time of each phase can be divided into several intervals as shown in Figure 2.26. t_G^0 is the beginning time of current green phase, T_G^{Ini} is the initial green time which intents to provide sufficient green time to clear at least one vehicle stored between the advanced detector and the stop line, T_G^{Last} is the maximum green time that a vehicle can pass the stop line without signal delay, U is the unit extension green time, $G_{\min}$ and $G_{\max}$ are minimum and maximum green time respectively.

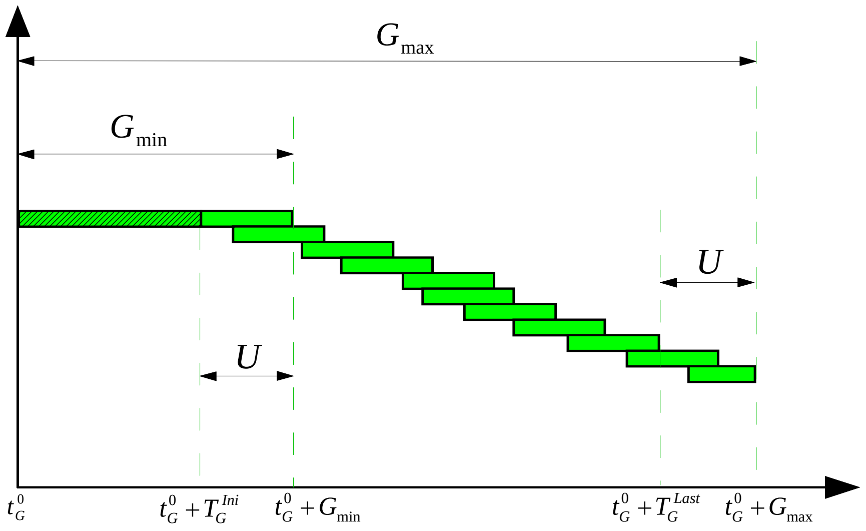
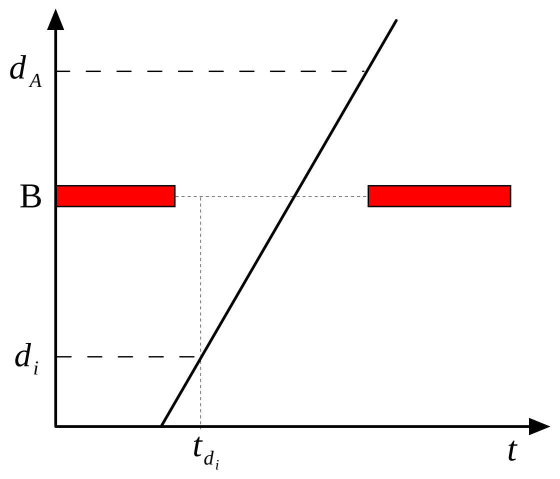


Figure 2.26: Time Intervals along Green Phase Horizon

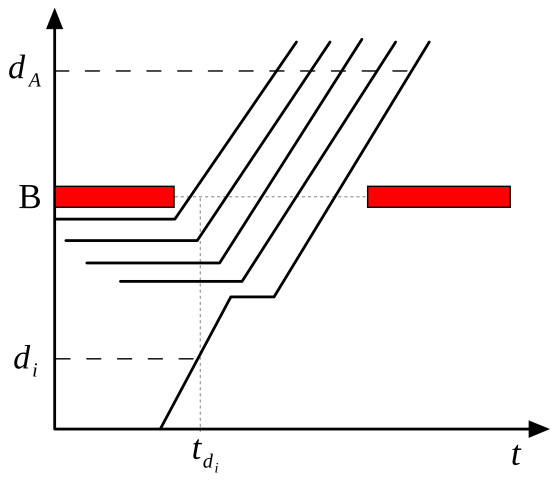
As shown in Figure 2.27, the green phase actuation model consists of mainly four cases based on the time interval when vehicle arrives at the advanced detector d_i. Figure 2.27 (a) shows the situation when a vehicle arrival event is trigged in the green phase before $t_G^0 + T_G^{Last}$, the vehicle can pass the intersection within initial green or several additional extension time U without any delay. To be noted that no standing queue will occur here since it is assumed the initial green interval T_G^{Ini} will clear all vehicles potentially stored between the detector and the stop line. In Figure 2.27 (b), the vehicle also arrives at the time before $t_G^0 + T_G^{Last}$, but with standing queue ahead. When a vehicle arrival event occurs after time $t_G^0 + T_G^{Last}$ as shown in Figure 2.27 (c), the green time will end after $t_G^0 + G_{\max}$ even under actuated signal control because of green out. In such case, the detected vehicle must wait until next green phase to pass the intersection. When a vehicle arrives after $t_G^0 + T_G^{Last}$ with queue ahead, the delay time T_q is added as depicted in Figure 2.27 (d).

The detailed calculation equations for the four cases are also depicted in Figure 7.3. $T_{d_i - S_B}$ denotes the travel time from detector d_i to stop line at intersection B, and $T_{S_B - d_A}$ denotes the travel time from the stop line at intersection B to detector d_A at intersection A. It is assumed both $T_{d_i - S_B}$ and $T_{S_B - d_A}$ are free flow travel time which can be roughly calculated according to the speed limit and travel distance. However, the typical travel speed is slightly higher than the speed limit when traffic is light. Studies from recent research found that the vehicle speed on arterials tends to be constant for un-congested traffic (Lin et al. 2003). Therefore it is reasonable to use the average travel time speed measured at off-peak hours to calculate the "free flow travel time" in this research.



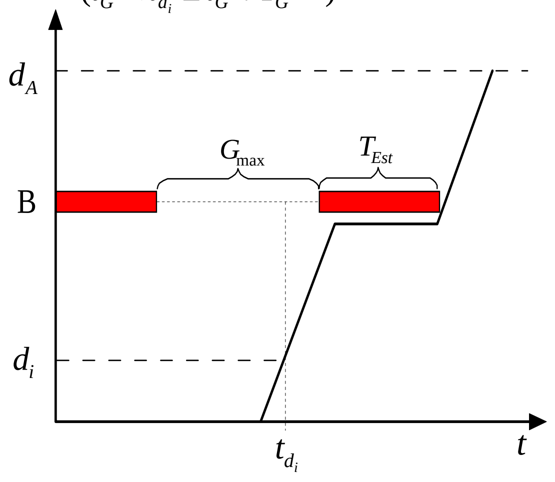
(a) Minimum Green & Extension Green trigged event: No Delay

$$t_{d_A} = t_{d_i} + T_{d_i - S_B} + T_{S_B - d_A}$$
$$(t_G^0 < t_{d_i} \le t_G^0 + T_G^{Last})$$



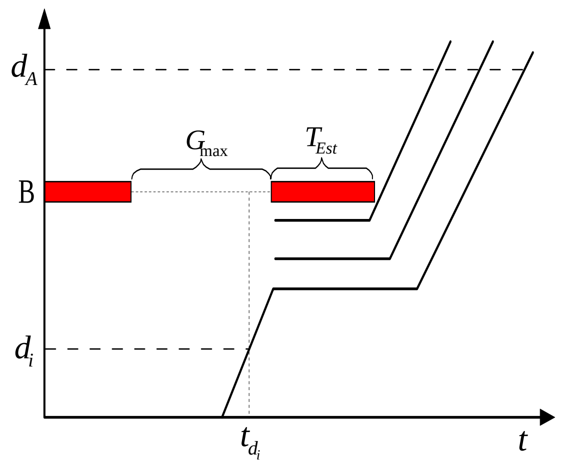
(b) Minimum Green trigged event: Queuing Delay

$$t_{d_A} = t_{d_i} + \max\{T_q, T_{d_i - S_B}\} + T_{S_B - d_A}$$
$$(t_G^0 \le t_{d_i} \le t_G^0 + T_G^{Last})$$



(c) Extension green trigged event: Signal Delay

$$t_{d_A} = t_G^0 + G_{\max} + T_{Est.} + T_{S_B - d_A}$$
$$(t_G^0 + T_G^{Last} < t_{d_i} \le t_G^0 + G_{\max})$$



(d) Maximum Green trigged event: Signal & Queuing Delay

$$t_{d_A} = t_G^0 + G_{\max} + T_{Est.} + T_q + T_{S_B - d_A}$$
$$(t_G^0 + T_G^{Last} < t_{d_i} \le t_G^0 + G_{\max})$$

Figure 2.27: Green Phase Actuation Model

The queuing delay T_q is the time required to clear the queuing vehicles between the subject vehicle and the stop line, it can be estimated as Equation 7.1 based on the number of vehicles in the queue (McShane, 2004).

$$T_q = l_l + h \times n_q \tag{7.1}$$

where l_l is the start-up lost time, h is the saturation headway, and n_q is the number of vehicles in the queue, which can be calculated as the difference of the arrival counts and the departure counts. The arrival count can be measured by the advanced detector, but the true departure count is unknown without the stop line detector installed. However, the number of queuing vehicles can be roughly estimated as shown in Equation 7.2, assuming the queue will not back up over the advanced loop detectors.

$$n_q(\tau) = \begin{cases} \max\{0, \quad A(\tau) - D(t_G^0) - s \times (\tau - t_G^0)\}, & \text{when green is on} \\ \qquad A(\tau) - D(t_G^0 + g), & \text{otherwise} \end{cases} \tag{7.2}$$

where

$n_q(\tau)$ is the number of vehicles in the queue at time τ ;

$A(\tau)$ is the arrival count at time τ ;

t_G^0 is the start of the green time of current cycle;

s is the saturation flow rate;

g is the green time of the current cycle;

$D(t_G^0)$ and $D(t_G^0 + g)$ are the departure counts at the start of the current green, and the end of the current green, respectively.

2.7.3. Red Phase Actuation Model

Green phase actuation model gives out the predicted travel time for vehicles passing an advance detector during green time interval. If the vehicle passes the detector during red time interval, the model will be a slightly different since the vehicle needs to wait until the following green phase appears. The time interval from the event trigged to the end of the current red phase (or say the start of the next green phase) need to be estimated. For actuated signal control, the red time is difficult to be estimated directly. However, the red time for the subject movement is the summation of the green phases for all other movements at the intersection. Therefore, the red time can be measured by sum up the green time of all other conflicting movements. Assuming intersection *B* adopts an eight-phase dual-ring signal control as shown in Figure 2.28, and for instance ΦD is the green time interval for through movement vehicles passing detector d_i, then the maximum red time interval for the movement is the summation of green time intervals of all other movements as can be depicted by Equation 7.3.

$$T_{red}^{\Phi D} = T_{Green}^{\Phi A} + T_{Green}^{\Phi A_2 (or\ \Phi A_1)} + T_{Green}^{\Phi B} + T_{Green}^{\Phi C} + T_{Green}^{\Phi C_2 (or\ \Phi C_1)} \tag{7.3}$$

where $T_{red}^{\Phi D}$ is the red time of ΦD, $T_{Green}^{\Phi A}$ is the green time of ΦA, and so on. To be noted that, Equation 7.2 gives the maximum red time for vehicles arrival at the detector during time interval t_0 to t_1. If the vehicle arrivals during time interval t_1 to t_2 then $T_{Green}^{\Phi A}$ do not need to be considered since the vehicle is not delayed during that time interval. Similarly, $T_{Green}^{\Phi A} + T_{Green}^{\Phi A_2 (or\ \Phi A_1)}$ can be ignored if the vehicle arrives during interval t_2 to t_3, $T_{Green}^{\Phi A} + T_{Green}^{\Phi A_2 (or\ \Phi A_1)} + T_{Green}^{\Phi B}$ does not need to be count if the vehicle arrives during interval t_3 to t_4, and only $T_{Green}^{\Phi C_2 (or\ \Phi C_1)}$ need to be estimated if the vehicle arrives during interval t_4 to t_5.

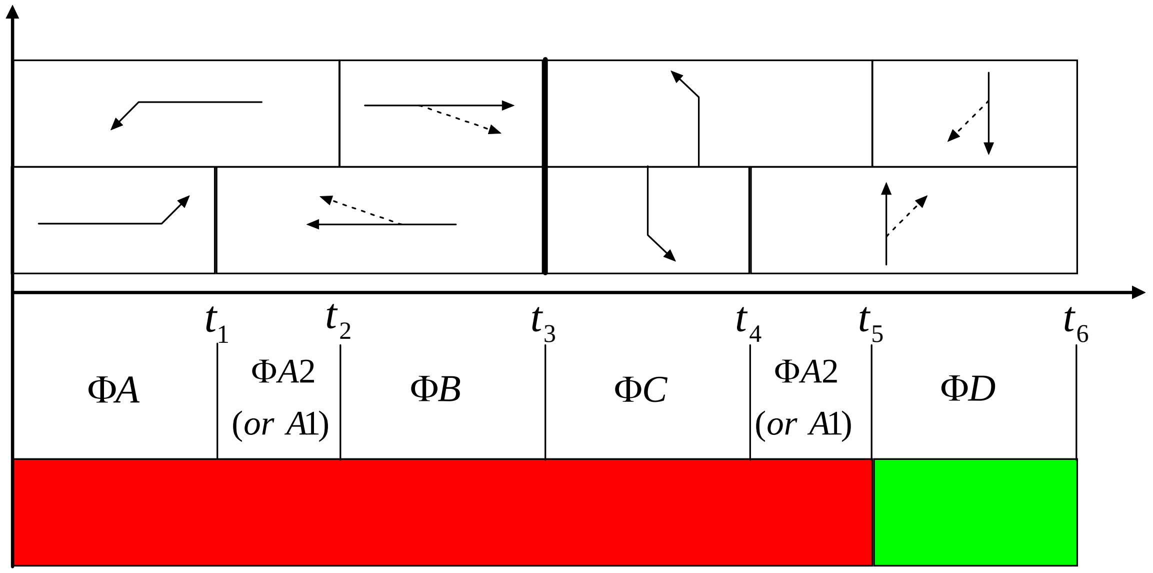


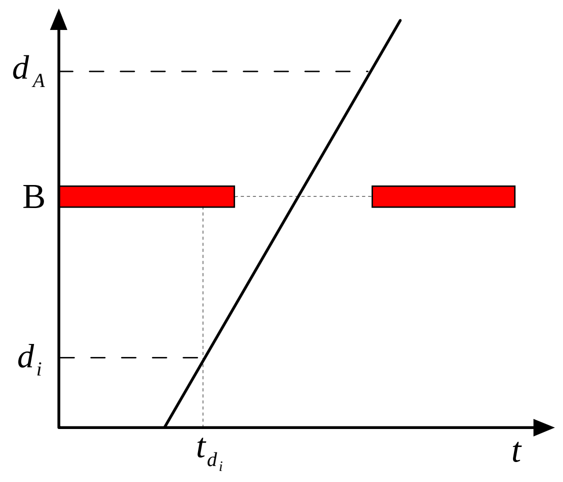
Figure 2.28: Dual-ring Control in Time Horizon

Equation 7.3 depicts that estimating the red phasing time is equal to estimating the green time of other phases. However, due to the property of actuation signal control, the actual green times are various values and can not be directly captured in advance. In this research, the time-series analysis is applied to forecast the "future" green times with historical data. It is assumed that the green time of each phase is correlated with the past and present ones.

Smoothing methodology is a time-series analysis technique that applied to provide a reasonably good short-term prediction. Two major types of smoothing techniques are tested in this research: a) simple moving average which assigns equal weights to past values to determine the moving average, and b) exponential smoothing which perceives the existence of random effects, and attempts to eliminate the irregularities from the underlying pattern (Washington et al. 2003). The exponential smoothing shows better results in the proposed model and thus be applied to predict the phase timing as Equation 7.4.

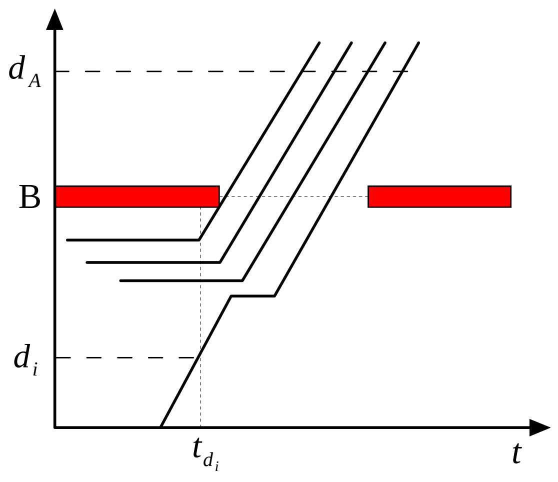
$$T_{i+1} = kT_i + k(1-k)T_{i-1} + k(1-k)^2 T_{i-2} + k(1-k)^3 T_{i-3} + \dots \quad (7.4)$$

where $k = 1/N$, and N is the number of previous values. For the testing example discussed later in this paper, N is selected with value 5, i.e. the phase time of last five cycles are used to predict the corresponding phase time of the present cycle.



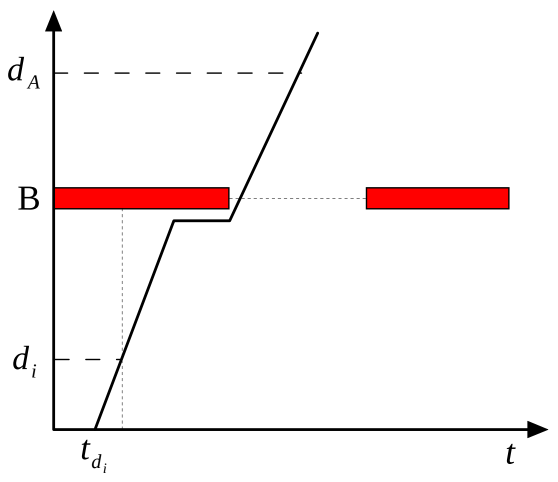
(a) Red trigged event: No Delay

$$t_{d_A} = t_{d_i} + T_{d_i - S_B} + T_{S_B - d_A}$$
$$(t_R^0 + T_R^{Wait} \le t_{d_i} < t_R^0 + \hat{T}_R)$$



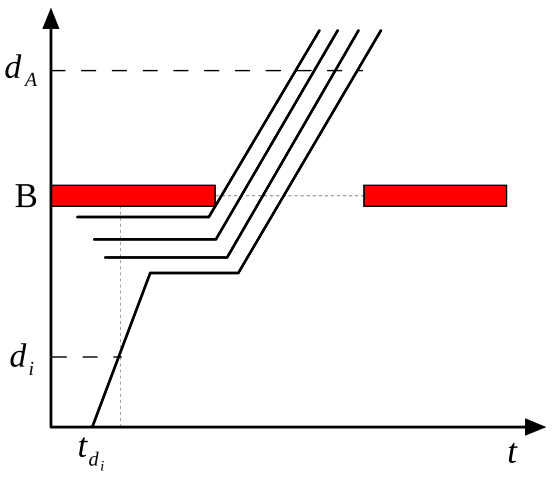
(b) Red trigged event: Queuing Delay

$$t_{d_A} = t_{d_i} + \max\{T_q, T_{d_i - S_B}\} + T_{S_B - d_A}$$
$$(t_R^0 + T_R^{Wait} \le t_{d_i} < t_R^0 + \hat{T}_R)$$



(c) Red trigged event: Signal Delay

$$t_{d_A} = t_R^0 + \hat{T}_R + T_{S_B - d_A}$$
$$(t_R^0 \le t_{d_i} < t_R^0 + T_R^{Wait})$$



(d) Red trigged event: Signal & Queuing Delay

$$t_{d_A} = t_R^0 + \hat{T}_R + T_q + T_{S_B - d_A}$$
$$(t_R^0 \le t_{d_i} < t_R^0 + T_R^{Wait})$$

Figure 2.29: Red Phase Actuation Model

Similar as the green phase actuation model, the red time trigged event can also be divided into four cases as shown in Figure 2.29. t_R^0 is the starting point of the current red phase. $\hat{T}_R$ is the maximum estimated red time for the current movement. T_R^{Wait} is the maximum waiting time for a vehicle suffered for the next green phase, which is equal to $\hat{T}_R$ minus the unit extension time *U*. Figure 2.29 (a) depicts the case when a vehicle arrival event is trigged in a red phase but the phase changes from red to green before it arrives at the stop line, then the vehicle can pass the intersection *B* without stop. Figure 2.29 (b) shows the similar scenario except the vehicle encounters additional standing queue. In Figure 2.29 (c) and Figure 2.29 (d), the vehicle arrival event is triggered before time $t_R^0 + T_R^{Wait}$, and the signal delay is counted with or without standing queue exists respectively.

Combining the above green phase actuation model and the red phase actuation model, the vehicle's arrival time at the subject intersection can thus be predicted for optimizing the signal control.

2.8 Dynamic Turning Proportion Prediction Model

2.8.1. Background

The turning proportion prediction is important for predicting traffic pattern at the subject intersection. The flow rate at intersection *A* can be concluded from the probabilities of the number of vehicles that pass an upstream detector (d_l, d_t, d_r), travel along the route *BA*, and arrive at the downtown intersection *A*. In Head's PREDICT algorithm, this uncertainty is mentioned by introducing a probability p_i^{BA} when computing the expected number of vehicles from detector d_i to detector d_A at future time t_{dA}. However, the author does not give the method to obtain an exact p_i^{BA} value.

The topic about turning proportion estimation has been broadly explored and many models were proposed accordingly. Most of the models, such as those reviewed by Maher (1984), require only counts for one cycle but need prior turning proportion estimation. The accuracy of these methods highly depends on how well the prior estimates match the actual turning proportions. On the other hand, time-series methods do not need prior estimates but require a long time frame which impedes their responsiveness. These estimates become highly inaccurate during times of sudden and highly irregular turning movement changes caused by unforeseen events such as traffic accidents, in which they are needed mostly by the traffic signal system. Another method which estimates intersection turning movement proportions from less-than-complete sets of traffic counts was proposed by Davis and Len (1995). In their method both entering and exiting traffic counts at each of an intersection's approaches are used, in some occasion it even works under conditions that the number or placement of detectors does not support complete counting. However, the method is greatly restricted to geographic scenarios, if the detector configuration does not satisfy an identifiably condition, it fails to work. Chang and Tao (1997) give a time-dependent turning estimation for signalized intersections, in which by including the approximate intersection delay, the model can account for the impacts of signal setting on the dynamic distribution of intersection flows. To improve the estimation accuracy, some pre-estimated turning factions from a relatively long time interval need to be served as additional constraints for the estimation. Nobe developed four closed-form estimation methods in his dissertation (1997), maximum entropy (ME), generalized least-squared (GLS), least-squared error (LS) and least-squared error/generalized least-squared error (LS/GLS), which either need prior turning proportion estimates or require counts for three cycles. Despite several layouts are considering, all of the four models have a fixed two-phase cycle, and the right turning movements in each phase are always set as protected. In fact, most of the right turning in signal intersections work as permitted, and serve as minor movements throughout whole cycle.

Given that the entering vehicle counts and exiting vehicle counts during every traffic signal phase in each direction of an intersection, a simple estimation of turning proportions which relies only on short-term vehicle counts is proposed in this research for the general actuated signalized intersection.

Figure 2.30 shows the typical movements of eight-phase actuation controller. The exiting and entering volume counts are collected to estimate the turning proportion. It is assumed that the left-turn movements and through movements have protected phase timing, while right-turn movements are all permitted during the whole cycle. The westbound (Approach 3 in the figure) is selected as a demo approach to depict the proposed model. Other approaches can be examined with the same logic.

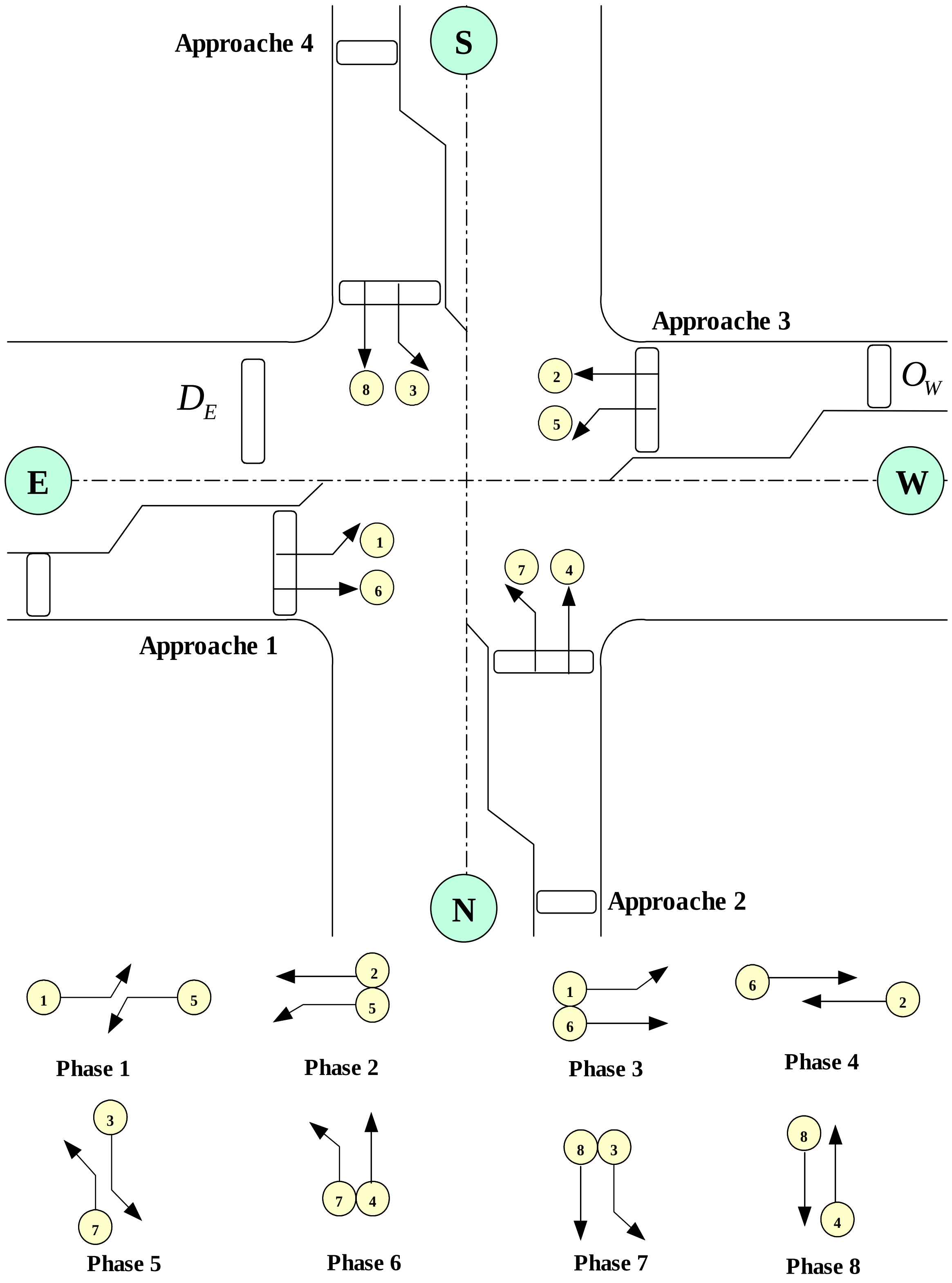


Figure 2.30: Typical Movements in Eight-phase Actuation Controller

2.8.2. Turning Proportion Model

The notations that will be used in the analysis of the turning proportion model described as follows.

O_i is the number of vehicles that entering from leg *i* during a cycle;

$D_j(l)$ is the number of vehicles that existing to leg *j* during phase *l*;

$V_{ij}(l)$ is the number of vehicles that entering from leg *i* and existing to leg *j* during phase *l*;

P_{ij} is the probability for vehicles entering from leg *i* and existing to leg *j*.

where

$i = \text{E, W, N, S}$

$j = \text{E, W, N, S}$

$l = 1, 2, 3, 4, 5, 6, 7, 8$

The known conditions related to the westbound movements can be described from Equation 8.1 to Equation 8.8 as shown below.

$$V_{WS}(2) = D_S(2) \tag{8.1}$$

$$V_{WS}(4) = D_S(4) \tag{8.2}$$

$$V_{WS}(5) = D_S(5) \tag{8.3}$$

$$V_{WS}(7) = D_S(7) \tag{8.4}$$

$$V_{WE}(2) + V_{SE}(2) = D_E(2) \tag{8.5}$$

$$V_{WE}(4) + V_{SE}(4) = D_E(4) \tag{8.6}$$

$$V_{WN}(1) + V_{SN}(1) = D_N(1) \tag{8.7}$$

$$V_{WN}(2) + V_{SN}(2) = D_N(2) \tag{8.8}$$

Equation 8.1 to Equation 8.4 describes that the right-turn movement from westbound to southbound V_{WS} has no conflicting movements during phases 2, 4, 5, and 7, so the destination detector count D_S is equal to V_{WS}. Phase 2 and phase 4 are the protected phases for the through movement V_{WE} from westbound to eastbound; however, it is not the only movement to eastbound during the two phases: the right-turn movement V_{SE} from southbound to eastbound can not be ignored as depicted in Equation 8.5 and Equation 8.6. Similarly, phase 1 and phase 2 are the protected phases for the left-turn movement V_{WN} from westbound to northbound, and the right-turn movement V_{SN} from eastbound to northbound also needs to be considered. Equation 8.7 and Equation 8.8 represent the logic.

In the proposed model, the turning movements V_{ij} are unknown variables, and the destination detector counts D_j are also known. For each approach, there are eight equations can be written as shown above, so we totally have 32 independent equations at four approaches. For the actuation control logic shown in Figure 8.1, there are six non-zero movements, including two protected left/through movements and four permitted right-turn movements, in each phase. Therefore the unknown variables are 48, and 16 additional equations are needed to solve the problem. The so-called adjusted uniform pattern assumption is applied in this research to make up the 16 equations. It is assumed the right-turn movements in one cycle are continuous and uniform. The right-turn volumes in the phases with conflicting movements can be produced from the volumes in the phases that do not have any conflicting vehicles. For instance, the right-turn volume at westbound during phases 1, 3, 6, and 8, can be calculated from the volumes of the rest phases which can be directly measured with the Equation 8.9.

$$V_{WS}(l)\big|_{l=1,3,6,8} = \frac{V_{WS}(2)+V_{WS}(4)+V_{WS}(5)+V_{WS}(7)}{T(2)+T(4)+T(5)+T(7)} \times T(l)\big|_{l=1,3,6,8} \times \alpha \qquad (8.9)$$

where $T(l)$ is the time period durance for phase l, and α is the adjusted factor that will be discussed in the next chapter.

There are totally 16 equations for the all four approaches, so the 48 equations are obtained to solve the 48 unknown variables, and the turning movement volumes can be computed. Therefore the total volumes for left-turn movement, through movement and right-turn movement can be calculated by just summing up the number of vehicles in corresponding phases, and the turning proportion can be calculated from Equation 8.10 to Equation 8.11.

$$P_{WS} = \frac{\sum_{l=1}^{8} V_{WS}(l)}{O_W} \qquad (8.10)$$

$$P_{WE} = \frac{V_{WE}(2)+V_{WE}(4)}{O_W} \qquad (8.11)$$

$$P_{WN} = \frac{V_{WN}(1)+V_{WN}(2)}{O_W} \qquad (8.12)$$

In the estimation model, exit passage detectors are needed to get the correct exit volumes of each phase. However, most of the current urban intersections do not install exit detectors except in some special parts of California. In order to estimate the exit flow during each phase, the available downstream detectors in neighboring intersection are used, even though distance in between. The counts from the downstream detectors in each phase need to be adjusted with the travel time from the stop line detectors to the downstream detectors. Similarly, the travel time can also be estimated under off-peak hours.

2.9 Performance Test and Analysis

2.9.1. Example Problem

The proposed travel time prediction model was examined by computer simulation based on the microscopic simulator PARAMICS V5 developed by Quadstone (2005). An additional simulation plug-in for actuated signal control is added to ensure the functionality of actuation signal control (Liu et. al, 2001). From the APIs provided by the actuated signal plug-in, the previous and current signal status can be acquired, therefore the travel time for each objective vehicle can be predicted.

The simulated network is based on the Main Street network in Logan, Utah, which includes 5 signalized intersections as shown in Figure 2.31. Although the prediction algorithm is applied to a single intersection, it is necessary to simulate the area surrounding the studied intersection to ensure realistic traffic flows. To provide a desirable environment, it is assumed no interlink sources or sinks between the links that connect the intersections. For the purpose of preliminary examination, the simulation time is 75 minutes with its first 15 minutes used for warm-up.

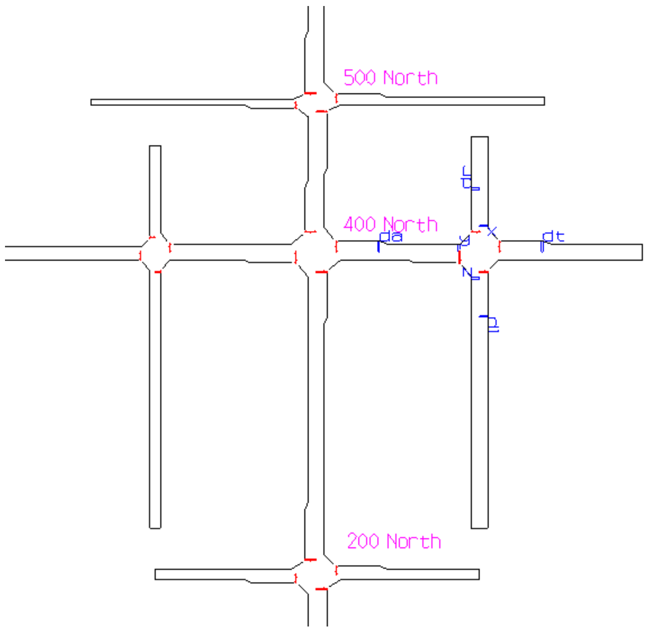


Figure 2.31: Example Network in Logan, UT

2.9.2. Adjusted Right-turning Factor Selection

As mentioned in Equation 8.9, the right-turn volume can be estimated with an assumption that the right-turn movement is in a continuously uniform pattern. It is a reasonable assumption that the right-turning vehicles continuously have the permitted phase timing in the whole cycle. However, during the phases with conflicting vehicles, the right-turn vehicles need to yield to the vehicles from other approaches due to the right-of-way. The right-turn flow during the phases with conflicting vehicles is usually less than the flow during phases without conflicting vehicles. Therefore, a adjust factor α between 0 and 1.0should be applied. We compared different scenarios with different k factors from 0.5 to 1.0. The Root Mean Squared Error (RMSE), a widely used error measure that can provide a fairly good initial estimate of the degree of fit between the simulated and the actual traffic measurements, is applied as Equation 9.1.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i)^2} \tag{9.1}$$

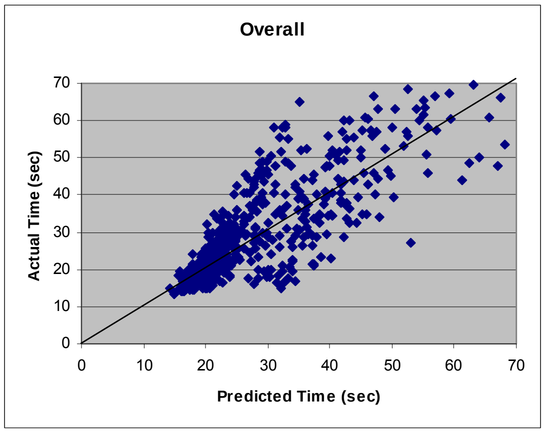
where *RMSE* is the root mean squared error, x_i is the i^{th} simulated traffic measurement value, and $\hat{x}_i$ is the i^{th} actual traffic measurement value. The result is listed in Table 2.32. From the table we can see *RMSE* has minimal values when the adjust *k* factor is equal to 0.75, which is thus selected in the proposed model.

Table 2.32: Adjusted Right-turning Factor Comparison

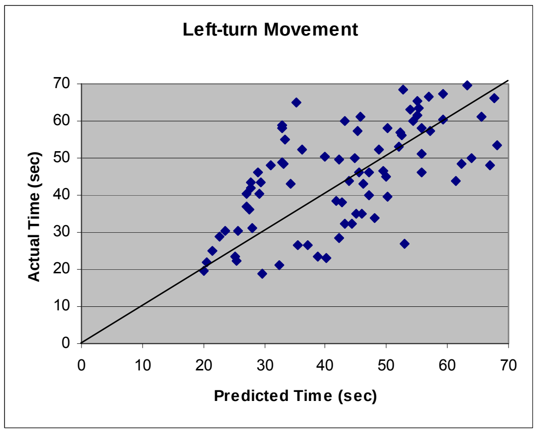
Adjust Factor	0.6	0.7	***0.75***	0.8	0.9	1.0
RMSE	1.040	1.000	***0.859***	1.024	1.100	1.435

2.9.3. Arrival Time Prediction Results

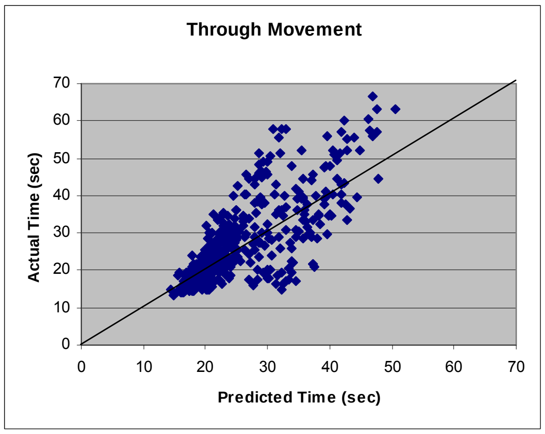
Figure 2.33 shows the plots of the actual travel time versus predicted travel time for the vehicles traveling from the upstream detector d_i to the downstream detector d_A during the simulation time. The perfect prediction line for each plot is along the line $y = x$, which is also the general trend for our plot points. Plot a) is the overall comparison for our travel time prediction model. Plots b) – d) show the comparisons according to the vehicle movements, i.e. left-turn movements, through movements, and right-turn movements. From the plots we can see that the prediction error from left-turn vehicles is a little bigger, which may due to the steering of left-turn movement is more complex: it usually involves with lane-changing and has larger turning-radius, thus harder to predict. The proposed model shows good prediction results for the through and right-turn movements. In the cases that vehicle's travel time is shorter, i.e. no signal or queue delay involved and vehicle can pass the intersection without stop, the prediction is quite close to the actual value. However, the error becomes larger as the travel time becomes longer, i.e. the vehicle may be delayed by signal or queue. It is reasonable since signal delay is difficult to accurately predict in actuation controller. Plots e) and f) depict the results from green actuation model and red actuation model separately. Most predictions of green actuation model is good, except for those vehicles do not catch their current green phase but need to wait for the next one.

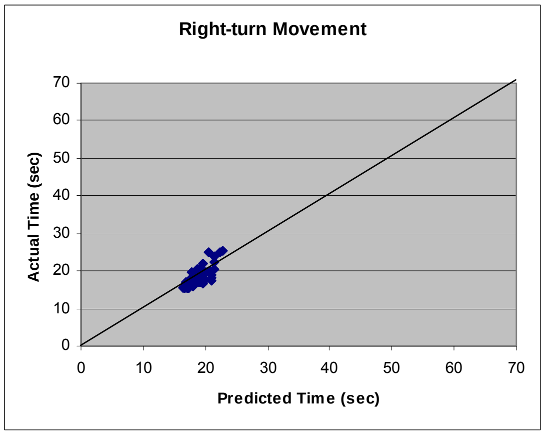


a)

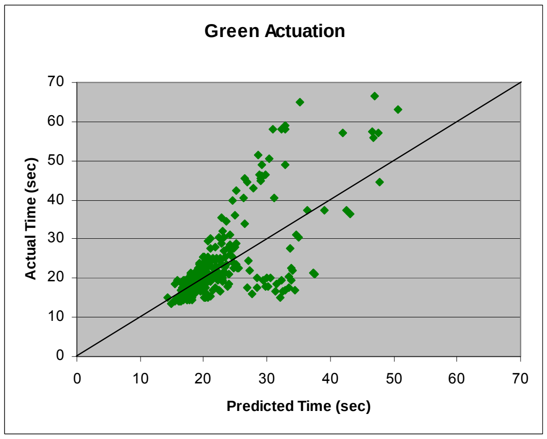


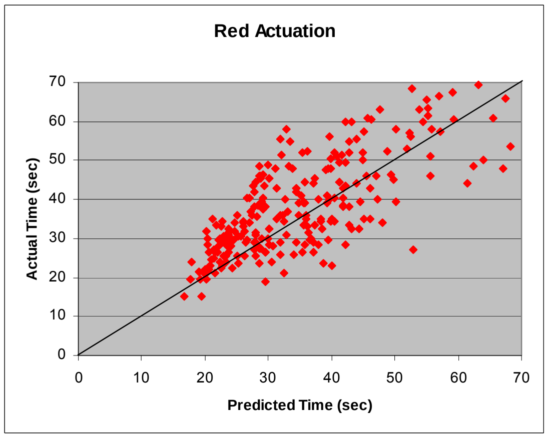
b)





c) d)





e) f)

Figure 2.33: Actual versus Predicted Travel Time

A slightly different statistic *RMSP* (Root Mean Squared Percent Error) as shown in Equation 9.2 is applied to verify the proposed model.

$$RMSP = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\hat{x}_i - x_i}{\hat{x}_i}\right)^2} \qquad (9.2)$$

The *RMSP*s of the example for overall, left-turn movement, through movement, and right-turn movement are 0.24, 0.32, 0.24 and 0.078 respectively. Most errors generated by the prediction model are in the acceptable range and the prediction results are reasonable and valuable.

2.9.4. Turning Proportion Results

Figure 2.34 shows the result of the turning proportion model proposed in this research. There are total 80 cycles implemented during the simulation time. The blue line shows the predicted number of the through movement vehicles that pass the upstream intersection *B* and arrive at the subject intersection *A*. The pink line shows the actual volume of the through movement from intersection *B* to intersection *A*. From the figure we can see that the model generates a quite good profile that matches the actual one. The statistic measure *RMSE* from Equation 9.2 is 0.859, which is acceptable.

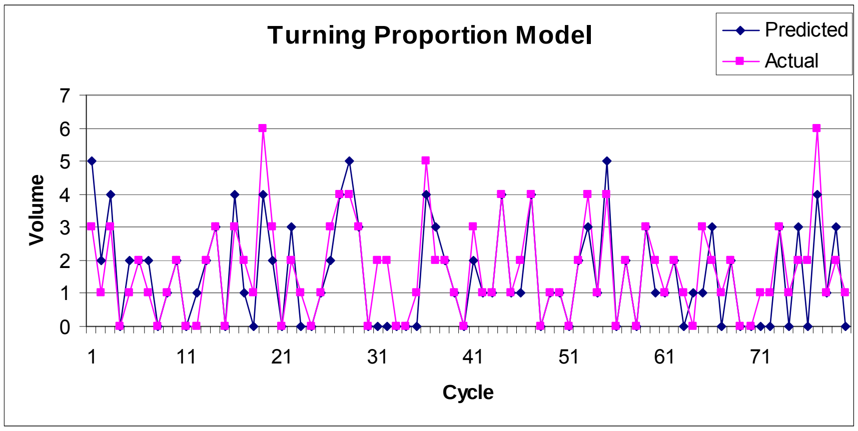


Figure 2.34: Actual versus Predicted Volume

2.10 Conclusions and Future Research

In the future, traffic signal systems will need to address many issues in the technical, social and political boundaries. The increase in urbanization and traffic congestion has challenged traffic engineers to manage the roadway systems with maximum safety and efficiency. As traffic volume continues to increase, roadway capacity will increase at a slower rate. New traffic control systems, such as adaptive traffic signals, will become increasingly critical for a city, county and the state to meet the growing transportation needs.

To deploy these advanced traffic systems in the field requires prior thorough testing in the laboratory. The Hardware-in-the-Loop Simulation has been a useful tool in evaluating the current adaptive signal controls in the laboratory. It has overcome the drawbacks and deficiencies in the software-only-simulation. By providing access to the traffic controller features and enhancing the advantages of the microscopic simulation, Hardware-in-the-Loop Simulation holds great promise for the future in transportation engineering.

The traffic flow prediction model proposed in this research is based on the data from the upstream actuated signal controllers and the surveillance systems. The model estimates the future traffic flow pattern thus can provide an optimized timing and phasing plan accordingly. By separating time horizon into two signal status: green phase and red phase, we explore each durance inclusively according to the vehicle arrival events. The prediction model can assist traffic controllers in responding to any sudden changes in traffic pattern and help to generate new optimal traffic management strategy without manual resetting. Different with some previous works, the proposed model fully utilize all available traffic information, not only the detector data but also the corresponding signal data, to do the forecasting. Moreover, no further investment is needed for additional equipments, only the data from existing installed hardware are used.

Although the simulation study of a simple testing environment presented in this section is limited, the results generated are reasonable and valuable. The following are the conclusions drawn from the project:

- Adaptive signal control performed better in higher demands compared to the actuated signal control and pre-timed signal control.
- Though the actuated signal control performed better in lower demands, the inclusion of dynamic signal optimization model and turn count estimation model may strengthen the performance of adaptive signal control in lower demands.
- The Hardware-in-the-Loop Simulation provided an excellent testing environment to test and evaluate different traffic control systems.

The contributions of this research for this project are summarized as follows:

- Propose an event-based short-term traffic flow prediction model and test with an arterial network that consists of 10 actuated signalized intersections.
- Implement the traffic flow prediction model and develop an online signal optimization module for an isolated intersection.
- Develop a testing environment using Hardware-in-the-Loop system to evaluate the performance of adaptive control system.
- As an outcome of this project a Paramics interface tool for HiLS is developed. This tool could be also used for educational purpose for better understanding of different signal controls.

The work summarized in this section shows that adaptive control signals would be an important strategy deployed in the urban areas in the future. In addition, Hardware-in-the-Loop Simulation provides a real time simulation environment to test traffic signal controls and adds value to the microscopic simulation. However, there is much more work to be done in these areas. The following discussion identifies several areas of research that would contribute in a positive way to the profession.

The current hardware-in-the-loop set up requires a pair of CID and a traffic controller to test signal control logic on a single intersection. For testing multiple intersections, multiple pairs of hardware's are required. The whole set should also be run in real time, thus making HiLS slower compared to software-only simulation. Developing traffic control software, including all the advanced features available in the actual controller into an application program interface would eliminate the CID and the actual traffic controller from the current HiLS setup.

Another direction of the future studies is to put the predict model together with the Hardware-in-the-Loop Simulation for a large arterial network with actuated signal lights. The results from software-only simulation have shown the future traffic flow can be predicted in a relative good extent. Considering any laboratory model and project should progress toward actual field implementation, the software-only environment should be followed and extended to a Hardware-in-the-Loop Simulation in which the emulated traffic controller logic is replaced with a real controller (Lucas et al., 2000). It allows the using all of the controller's true functionalities instead of the limited subset of standard features that is supported by simulation package. In addition, a Hardware-in-the-Loop Simulation also provides an opportunity to experiment with the implementation of advanced features, such as signal coordination strategies and signal preemption capacities.

3. Hardware in the Loop with 170 and 2070 Controllers

3.1. Preliminary Investigations

The objective of Task Order 5311, as described in the proposal, was “to develop enhanced Paramics simulation tools with substantial control functions of 170/2070 controllers.” At UC Berkeley, several approaches were explored for doing this, with an emphasis on specific applicability to the California-approved 170 and 2070 traffic signal controller types

Work was done the first year of the project to integrate a Paramics simulation of a simple two intersection network with a single 170 controller through a standard serial port connection. These 170s tests used 170 controllers with special EPROMS developed by Kai Leung of the Caltrans Division of Operations. These modifications allowed virtual loop detector inputs from the Paramics simulation to be sent over a serial port. This work was originally done on Paramics version 4, and had to be modified in order to run plug-ins on Paramics version 5. While the Paramics modeler did not significantly change in the switch from version 4 to 5, the Paramics programmer changed significantly. Many new functions were added, and an entirely new naming convention was created

Figure 3.1: Richmond Field Station Intelligent Intersection

Investigations were then carried out to see whether it would be possible to develop code to run Paramics HiLS with a 2070 running Siemens SEPAC software (Siemens-ITS, 2005a,b), using NTCIP (NTCIP 2007) over a serial port to place a call on the appropriate phase. This would have used NTCIP modules already developed for reading signal state and placing calls on a phase (Dickey 2008), as part of other California PATH research at the Richmond Field Station (RFS) Intelligent Intersection (see Figure 3.1) to communicate with the SEPAC software. However, although NTCIP allows placing a call on a phase, it does not allow virtual setting of detector inputs. This method would have involved emulating some of the 2070 logic in the plug-in, to convert the detection input created by the Paramics simulation into a call on the appropriate phase, and might not exactly mimic the real behavior of the controller with a detector input.

Making the same kinds of changes that were made by Kai Leung to the 170 to software running on a 2070 would have been a very desirable alternative means of HiLS. We originally hoped to be able to gain access to the 2070 Traffic Signal Control Program, used by Caltrans and originally written by Los Angeles Department of Transportation, and create a version that could be used for simulations. However, intellectual property concerns prevented us from getting access to this code base for development.

Subsequently, an open framework for traffic signal controller software has been developed at UC Berkeley (Zennaro 2008). This software has uniform interfaces for simulation and other external communication to the controller, including Paramics and Simulink integration libraries. The "Berkeley ATCP" control software based on this framework is now running on the 2070 in the RFS Intelligent Intersection and being used for testing of transit signal priority and traffic signal adaptation applications based on wireless communication (see Figure 3.2).

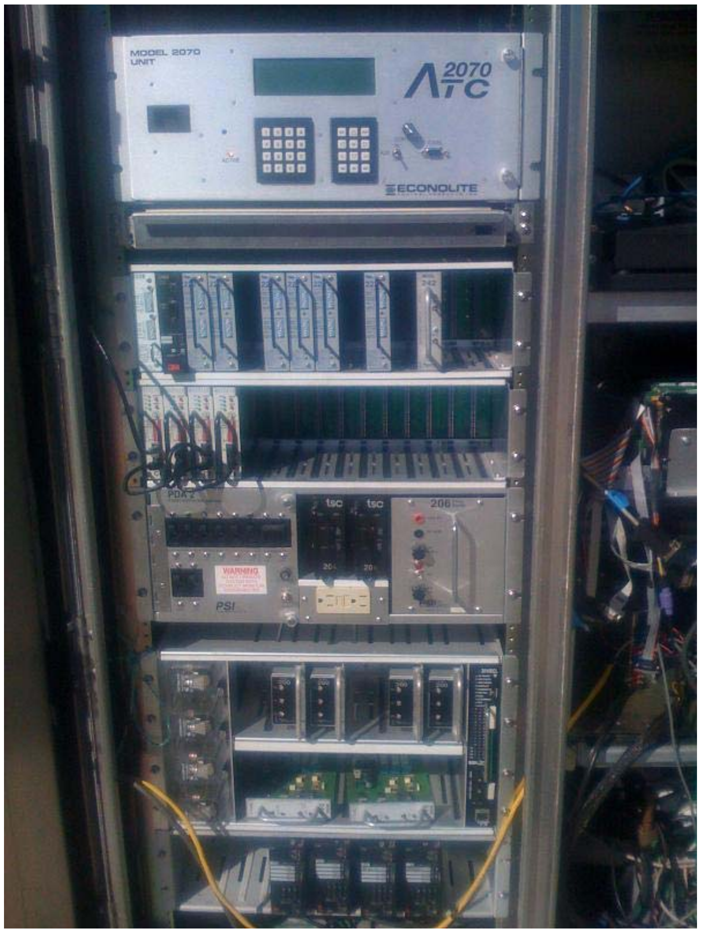


Figure 3.2: 2070 traffic controller at RFS Intelligent Intersection

3.2. Configuration of CID to work with 2070

In order to successfully communicate intersection inputs from a Paramics simulation through a McCain Traffic Supply/NIATT Controller Interface Device (CID) to an Eagle 2070 controller, a software plug-in was written for by the Paramics micro-simulation software and parameters within the default Siemens 2070 controller software had to be properly configured.

Small reconfigurations were needed for the Windows XP personal computer platform being used to execute the Paramics simulation.

Table 3:3: 2070 Detector Configuration

CID channel	2070 detector	C1 connector pin	170 function	170 phase
1	1	56	E&C	1
2	17	60	E&C	1
3	3	39	E&C	2
4	4	43	E&C	2
5	7	47	CALL	2
6	5	63	E&C	2
7	6	76	EXT	2
8	9	58	E&C	3
9	18	62	E&C	3
10	11	41	E&C	4
11	12	45	E&C	4
12	15	49	CALL	4
13	13	65	E&C	4
14	14	78	EXT	4
15	19	55	E&C	5
16	37	59	E&C	5
17	21	40	E&C	6
18	22	44	E&C	6
19	25	48	CALL	6
20	23	64	E&C	6
21	24	77	EXT	6
22	29	57	E&C	7
23	38	61	E&C	7
24	2,31	42	E&C	8
25	8,32	46	E&C	8
26	35	50	CALL	8
27	10,33	66	E&C	8
28	16,34	79	EXT	8
29	ped	67	ped6	2p
30	ped	69	ped7	4p
31	ped	68	ped8	6p
32	ped	70	ped8	8p

Once configured correctly, code was written in C to correctly send simultaneous inputs to the CID. The McCain Traffic Supply/NIATT CID was first connected to the Eagle 2070 traffic controller via the AMP C1 connector. The C1 connector diverges into 3 separate cables to be connected to the CID connector ports. The CID was then connected to a Windows XP PC through a standard USB cable. The CID drivers and software applications were then installed from the accompanying V2 XP installation disk.

After connecting and installing the CID to the PC, the “suitcase test” emulator was used to verify operation of the device. Since the 2070 controller option was grayed out (during start of suitcase emulator), the 170 controller option was used. Each channel was functional and mapped to specific detectors operating in a default 170 configuration. In this arrangement, each channel maps to a pin on the AMP connector, and this pin maps the signal to as an input for a specific detector. However, the McCain Traffic Supply documentation had no details regarding 2070 detector configuration and CID channels. The 2070 had 64 programmable detectors in default mode. Programming eight detectors at a time, after 10 trials the 2070 detectors were mapped to CID channels. The mapping is shown in Table 3.3.

Table 3:4: Detector and channel assignments used for tests

CID channel	2070 detector	Paramics Index	2070 detector operation mode	2070 phase	2070 EXT/10
1	1	10, 28	SBA	1	0
5	7	16, 17	SBA	2	0
7	6	19, 20	SBB	2	40
8	9	21, 22	SBA	3	0
12	15	1, 2, 5	SBA	4	4
14	14	3, 4, 27	SBB	4	50
15	19	18, 24	SBA	5	0
19	25	8, 9	SBA	6	0
21	24	6, 7, 29	SBB	6	40
22	29	25, 26	SBA	7	0
26	35	11, 12, 15	SBA	8	0
28	16,34	13, 14, 23	SBB	8	50

Once this mapping was completed, the 2070 detectors were programmed to the appropriate phase and function. Programming phases is straight forward via 3-Phase Data>8-Spec Detector>B-Contrl and setting the assigned phase. Programming detectors functions is more difficult. Referring to Table 1, each detector has a particular function: E&C, CALL, and EXT. The configuration used for this project was such that all detectors at the stop-bar are CALL, while all detectors upstream of the stop-bar are EXT. The Siemens default software allows configuration of each detector operation via 3-Phase Data>8-Spec Detector>B-Contrl and setting operation mode. Furthermore, the timing of each detector can be performed via 3-Phase Data>8-Spec Detector>B-Contrl and setting EXTEND or DELAY time. Once at the EPAC detector config data menu, one may configure the operation mode of each detector. All CALL detectors were configured to mode Stop Bar A (SBA), while all EXT detectors were configured to Stop Bar B (SBB). In SBA mode, the detector is active until a gap occurs, while in SBB mode the detector is detector is active until a gap great than EXTEND occurs. Please consult the SEPAC Training Guide (SEPAC-ITS, 2005b, p. 111) and the SEPAC Reference Manual (SEPAC-ITS, 2005a, p. 32) for more details.

Now that each phase has an assigned detector and operation mode, one must analyze the network to be simulated to assign Paramics detectors to phases. Following documentation of preparing network configuration for use with UC Irvine Actuated Signal Control plug-in (Chu 2003), stopbar and extension detectors for each phase can be quickly identified. Next, each

Paramics detector must have its index matched to the appropriate CID channel. This can be done by comparing the channel phase/operation with the Paramics detector phase/operation. The detector and channel assignments used for tests are shown in Table 3.4.

After configuring the 2070 detectors, the signal timing plan was programmed into the controller. Vehicle recalls and pedestrian times were also included.

Next, the plug-in (written in C) enabled Paramics to send simulation data via the CID buffer. A previous plug-in (Sullia, 2005) from Utah State University sets one channel during each time-step. One flaw with this technique occurs during periods with high saturation ratios (large flows of vehicles). The specific function used in this previous plug-in could only activate one channel per time-step. So, a new plug-in was written to activate multiple CID channels simultaneously. Each CID channel is activated for a time declared by the user. Channel activation was based on vehicle actuation of detectors. Source code was written using values from Table 3.4. The code reads Paramics detectors, sets appropriate values within temporary buffers, converts temporary buffers for the CID buffer, and writes the CID buffer with desired user timings.

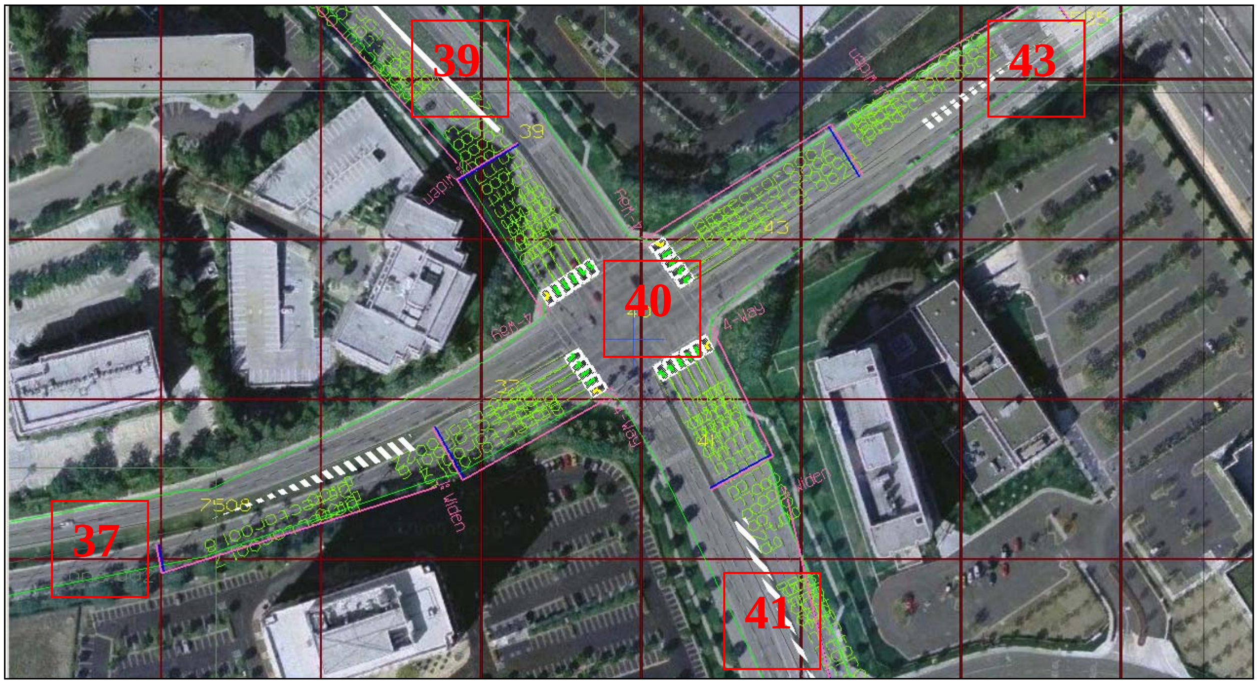


Figure 3.5. Simulated Intersection (nodes noted in red)

3.3. Comparison of 2070 HiL and UC Irvine Actuated Signal Control

Using the Hardware-in-the-Loop (HiL) plug-in, many simulations of the intersection illustrated in Figure 3.5 were performed and then compared to the *UC Irvine Actuated Signal Control plug-in*.(Liu 2003) Demands were set to three scenarios: full saturation from all approaches, full saturation for two approaches, and moderate demand on all approaches. The results were then compared to verify simulation accuracy of the software plug-in.

The HiL plugin and *UC Irvine Actuated Signal Control plug-in* were each used for 15 simulation runs with three demand settings. The seed number within Paramics Modeller was set to 0 (random), and each simulation was set to run for 1 hour and 15 minutes. The first 15 minutes was used as a warm up period, and data from 0:15 to 1:15 time values were used. Flow demands set are shown for reference in Table 3.18. Results were reviewed with Paramics Analyzer. Averages were obtained for each demand by averaging the four 15-minute values. Therefore, each simulation produced an average value of measurement.

Using a 95% confidence interval, dataset variances were compared using an F-test. Afterwards, a two-sample t-test (Table 3.6) was used based on the variance equality outcome. The values (rounded to 4 decimal places for readability) show the probability of the two means being equal when using a two tailed test. Table 3.7 and Table 3.8 summarize the differences found between the HiL and UC Irvine Actuated Signal Control plug-in (HiL data was used as a baseline). In these tables, and in the graphs in Figures 3.9-3.17, it can be seen that in these tests there were not any substantial difference in average parameters between HiLS and the software plug-in, even if there were not enough simulation runs to make a statistically significant conclusion.

Table 3:6: Hypothesis test of mean value comparison

Demand	Measurement	Approach (see Figure 3.5)			
		39-40	**43-40**	**37-40**	**41-40**
Fully Saturated	Avg. Delay	0.0000	0.0122	0.0508	0.0000
	Avg. Stoptime	0.0005	0.0000	0.0033	0.0000
	Avg. Flow	0.2166	0.0000	0.0000	0.1045
Barranca Saturated	Avg. Delay	0.0357	0.0000	0.0000	0.6276
	Avg. Stoptime	0.0417	0.0001	0.0000	0.9904
	Avg. Flow	0.1367	0.0218	0.0000	0.0692
Spread Demand	Avg. Delay	0.0009	0.0000	0.0000	0.0030
	Avg. Stoptime	0.0016	0.1216	0.0039	0.0008
	Avg. Flow	0.2590	0.7030	0.6617	0.8699

Table 3:7: Percentage difference between plug-ins

Demand	Measurement	Approach			
		39-40	**43-40**	**37-40**	**41-40**
Fully Saturated	Avg. Delay	11.68	2.77	2.01	12.90
	Avg. Stoptime	3.70	7.88	3.27	5.17
	Avg. Flow	0.77	10.28	4.10	1.61
Barranca Saturated	Avg. Delay	3.24	22.62	16.08	0.83
	Avg. Stoptime	3.42	5.85	5.63	0.03
	Avg. Flow	2.87	1.27	3.89	3.32
Spread Demand	Avg. Delay	9.83	15.70	15.86	9.06
	Avg. Stoptime	5.02	2.40	4.15	6.34
	Avg. Flow	1.57	0.42	0.53	0.16

Table 3.8. UC Irvine Actuated Signal Control plug-in values as a percentage of HiL values

Demand	Measurement	Approach			
		39-40	**43-40**	**37-40**	**41-40**
Fully Saturated	Avg. Delay	88.32	97.23	97.99	87.10
	Avg. Stoptime	96.30	92.12	96.73	94.83
	Avg. Flow	100.77	110.28	104.10	101.61
Barranca Saturated	Avg. Delay	96.76	77.38	83.92	99.17
	Avg. Stoptime	96.58	94.15	94.37	100.03
	Avg. Flow	102.87	101.27	103.89	96.68
Spread Demand	Avg. Delay	90.17	115.70	115.86	90.94
	Avg. Stoptime	94.98	102.40	104.15	93.66
	Avg. Flow	98.43	100.43	99.47	99.84

Figure 3.9: Comparison of Average Delay (Saturated) for each Approach

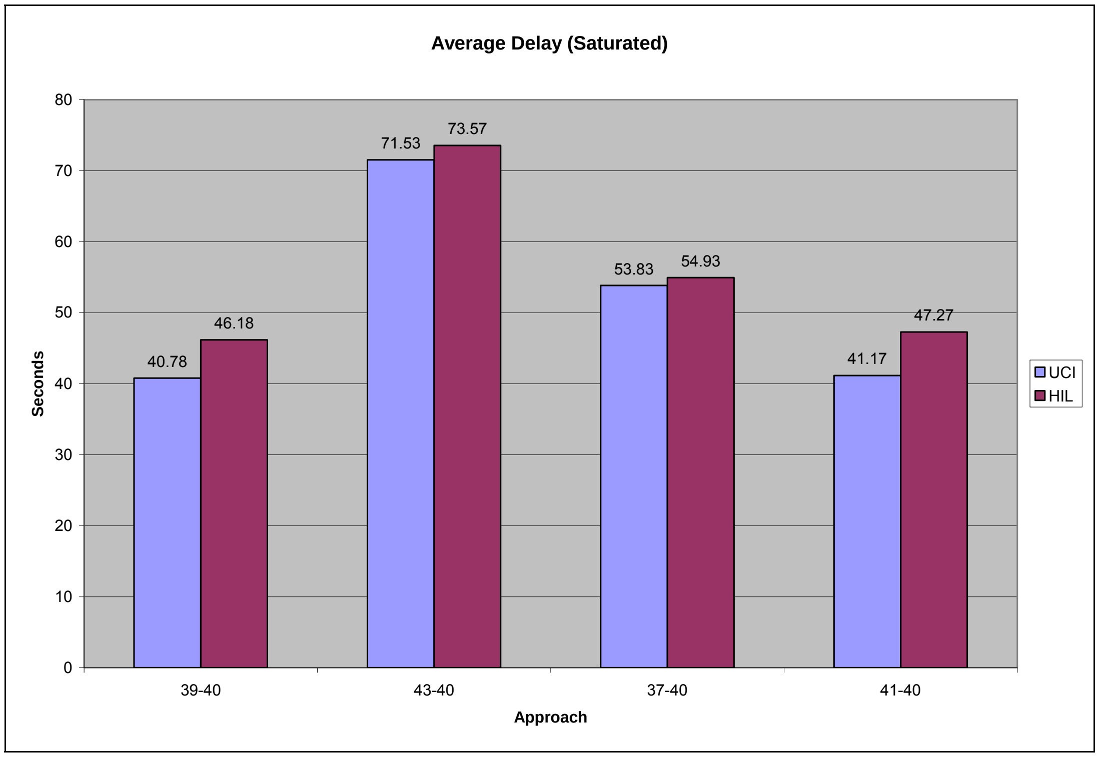


Figure 3.10: Comparison of Average Stoptime (Saturated) for each Approach

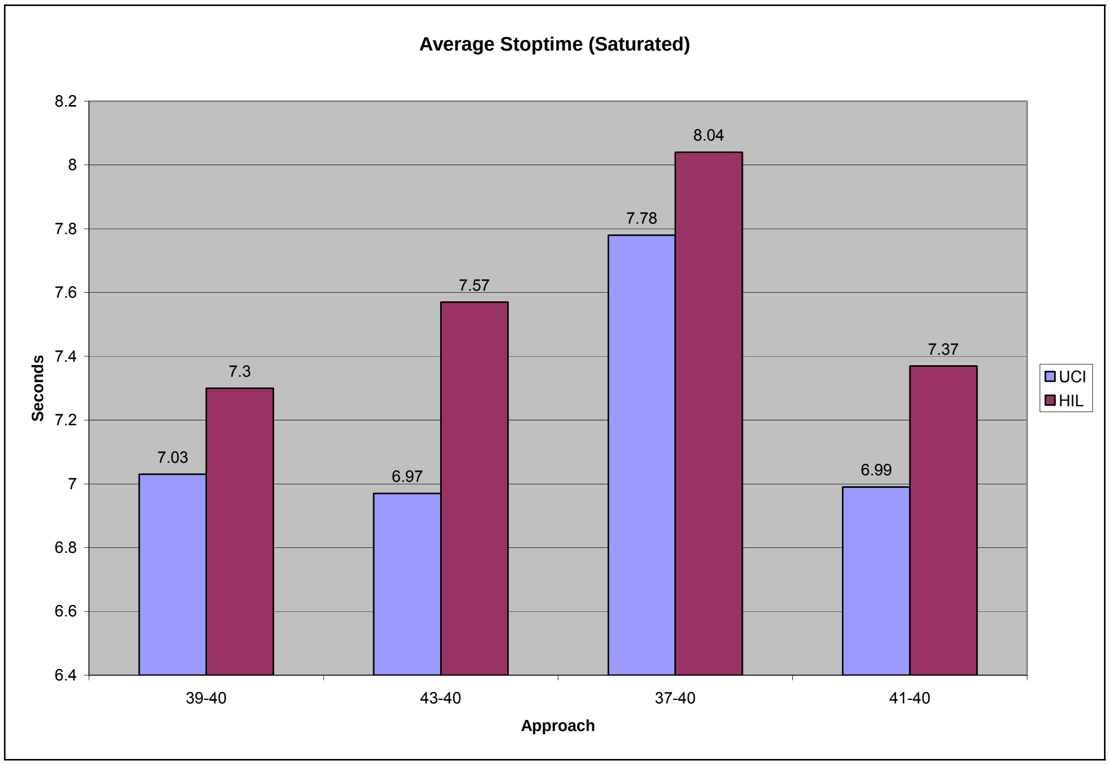


Figure 3.11: Comparison of Average Flow (Saturated) for each Approach

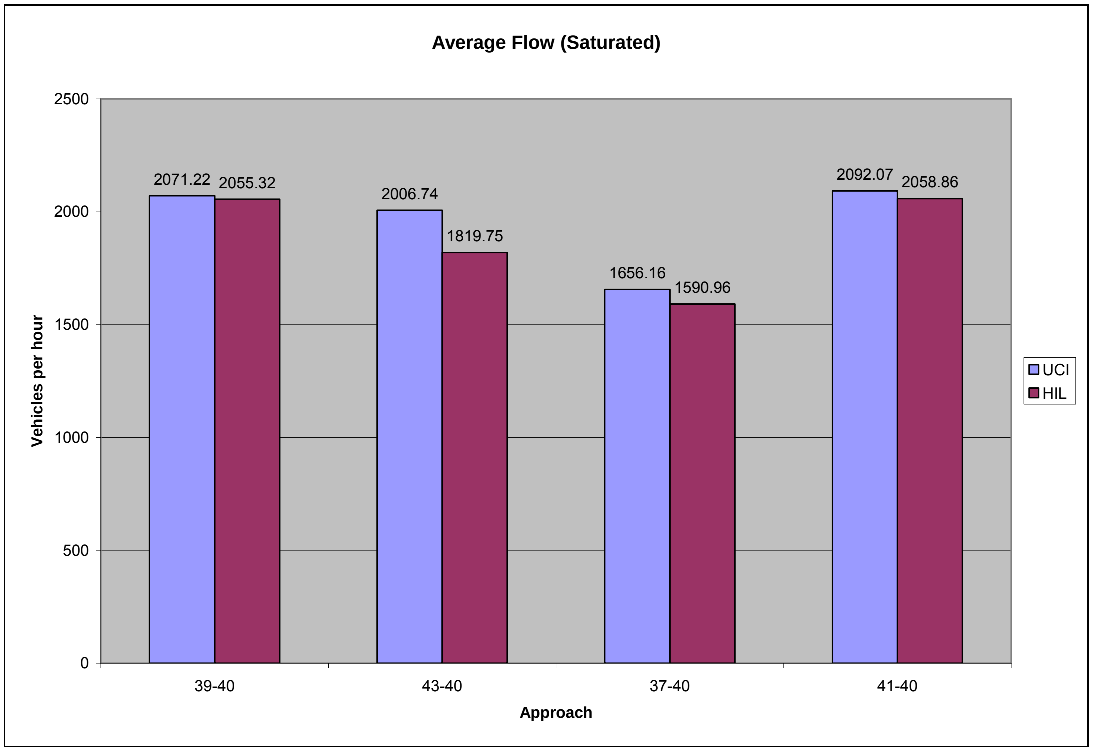


Figure 3.12: Comparison of Average Delay (Barranca Saturated) for each Approach

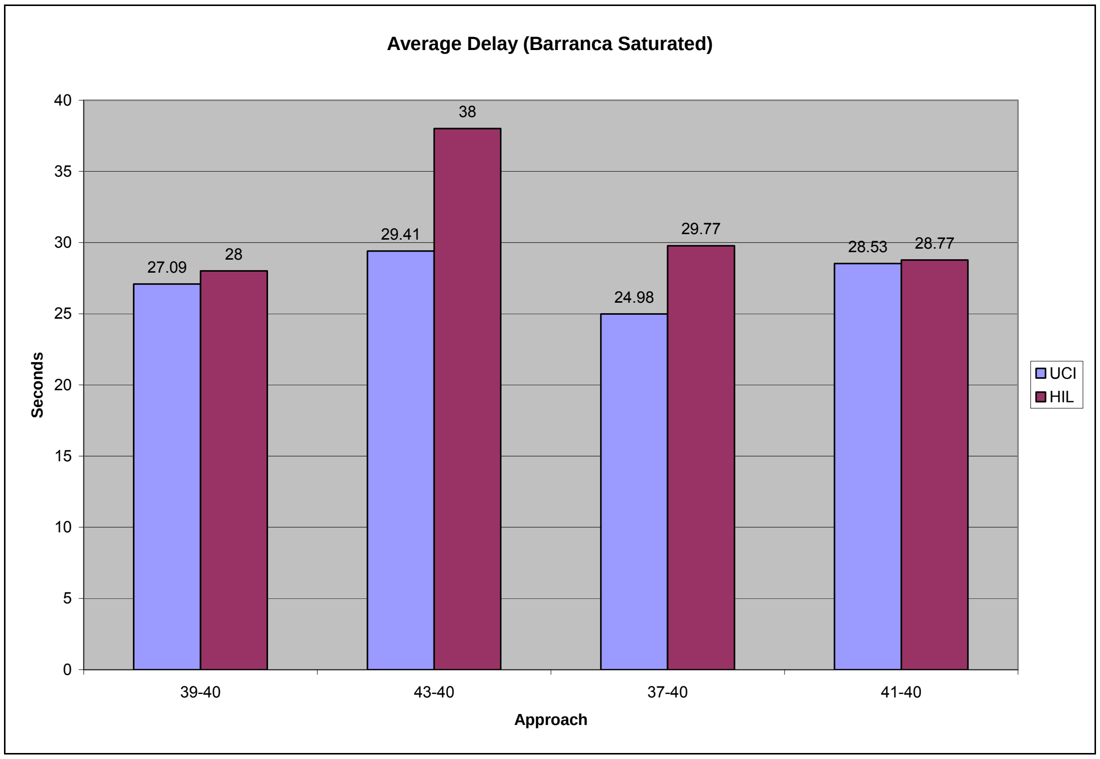


Figure 3.13: Comparison of Average Stoptime (Barranca Saturated) for each Approach

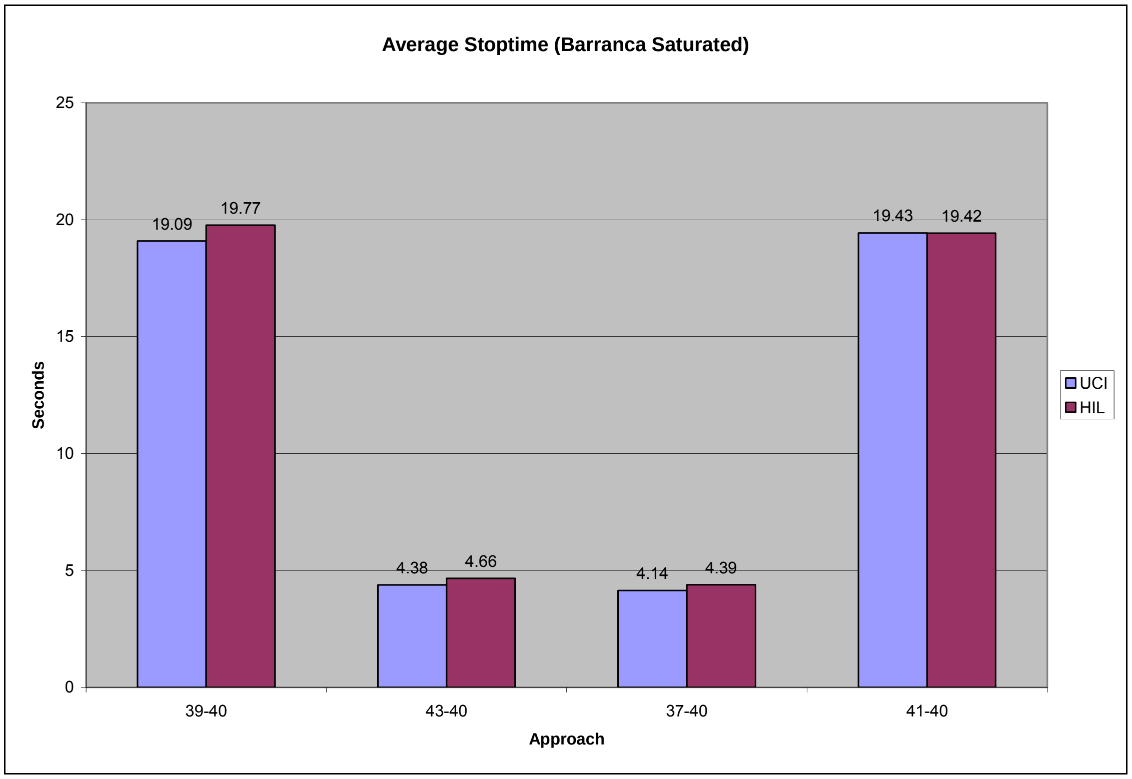


Figure 3.14: Comparison of Average Flow (Barranca Saturated) for each Approach

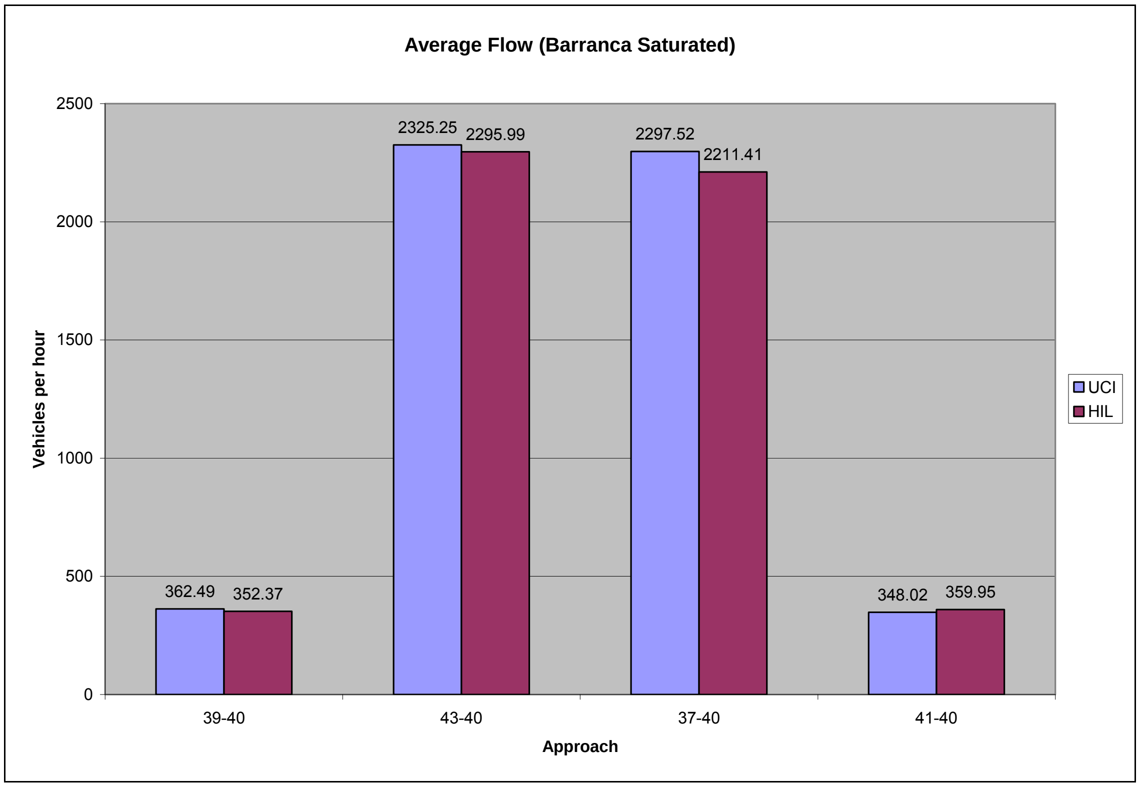


Figure 3.15: Comparison of Average Delay (Spread Demand) for each Approach

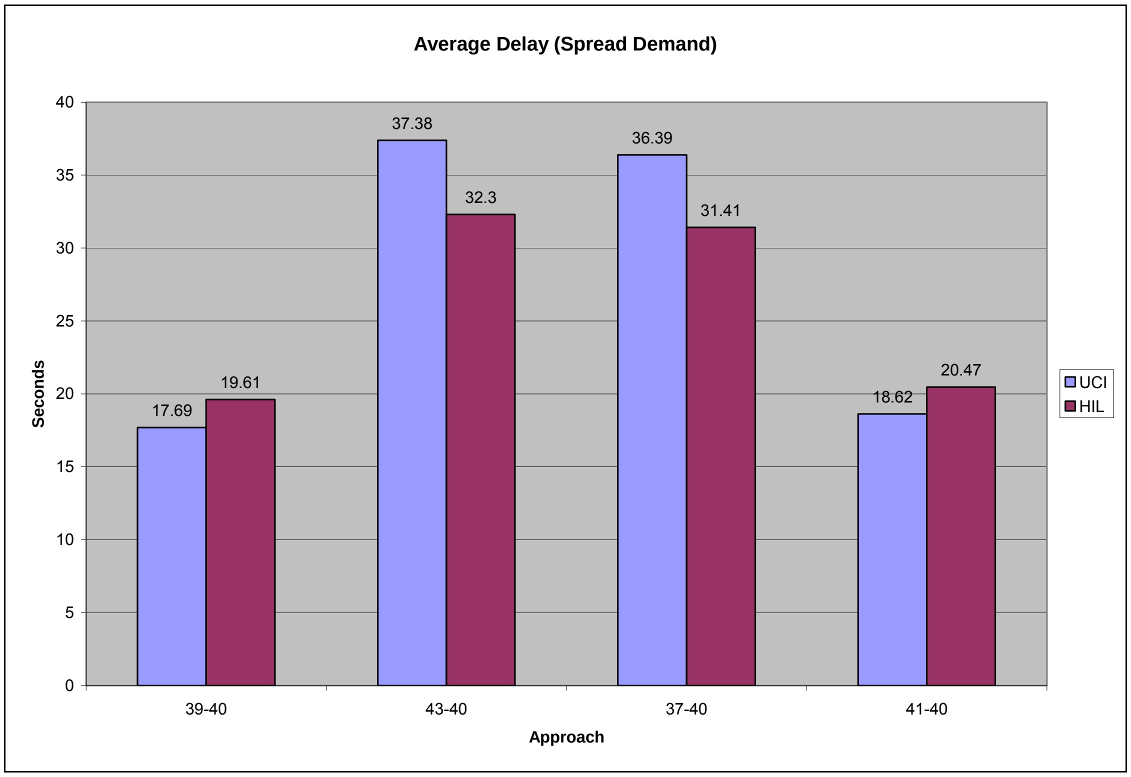


Figure 3.16: Comparison of Average Stoptime (Spread Demand) for each Approach

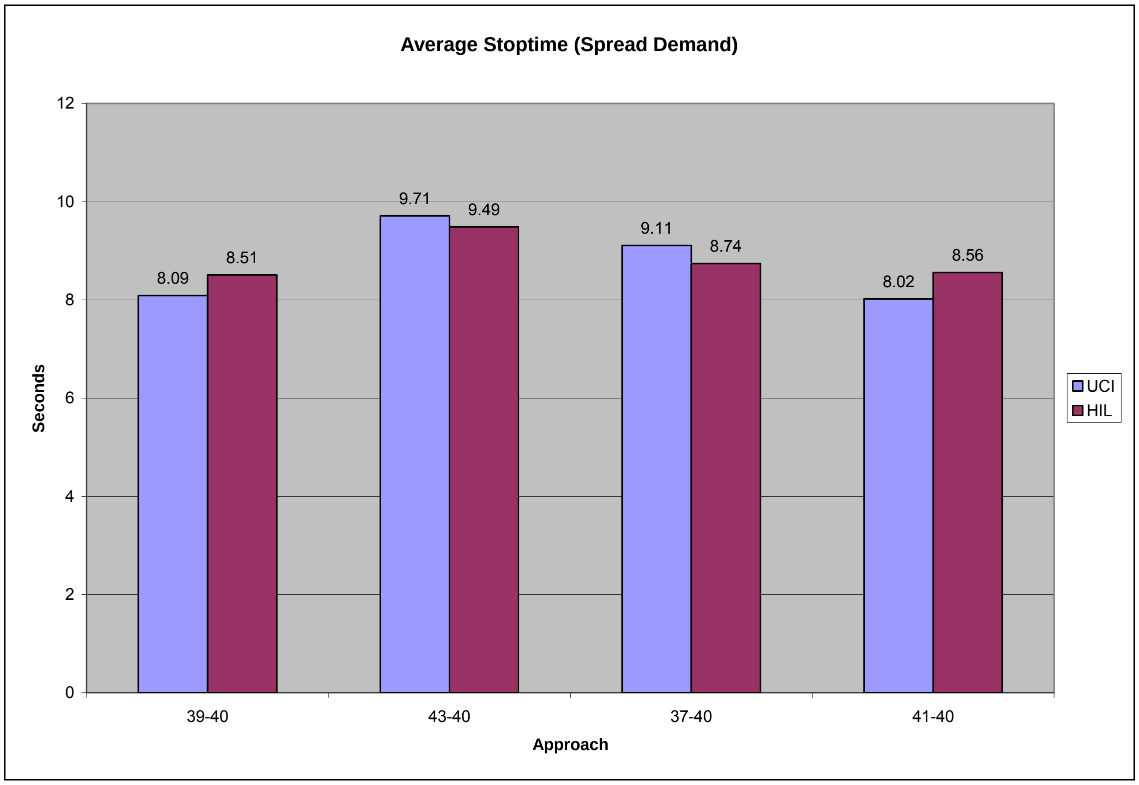


Figure 3.17: Comparison of Average Stoptime (Spread Demand) for each Approach

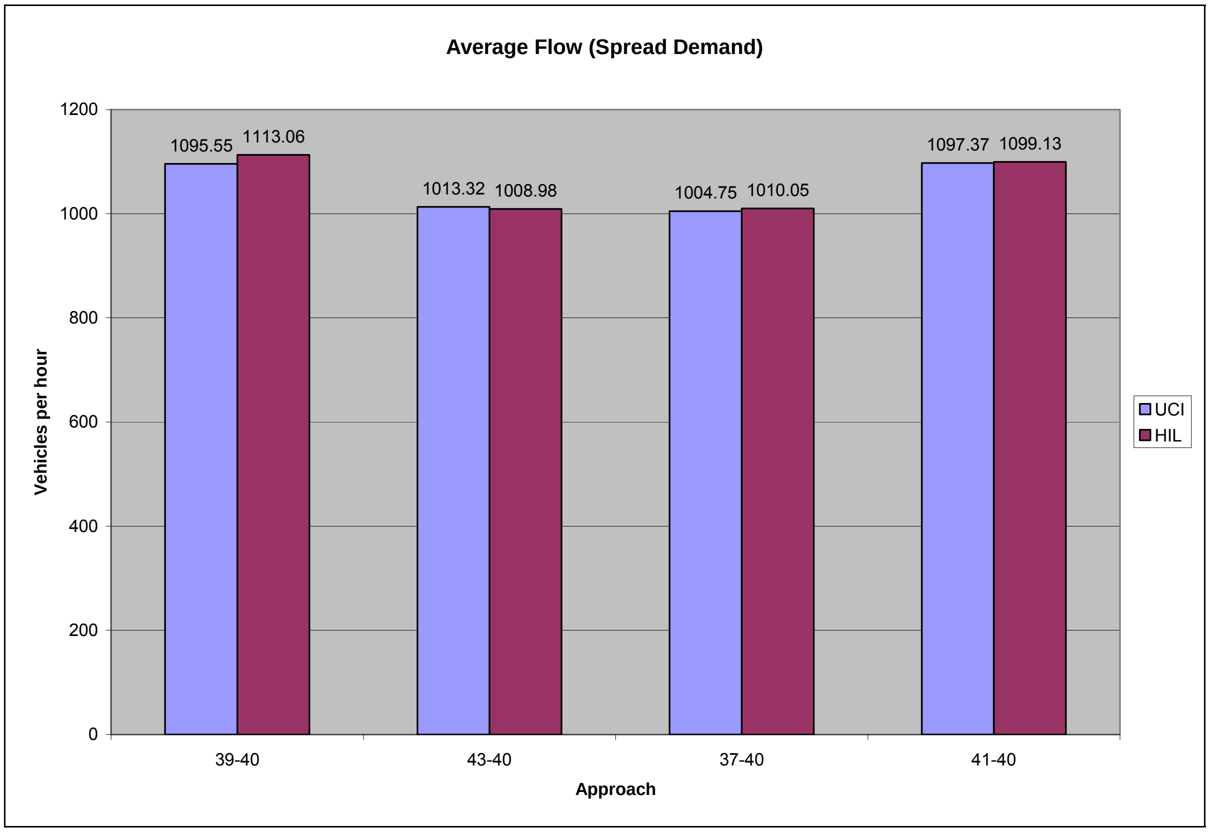


Table 3:18 Flow Demands set for each scenario (flow from node at left to node at top of column)

	Fully Saturated				Baranca Saturated				Spread Demand			
	43	**37**	**39**	**41**	**43**	**37**	**39**	**41**	**43**	**37**	**39**	**41**
43	0	1400	400	400	0	1500	400	400	0	1500	400	400
37	1200	0	300	300	1500	0	400	400	1500	0	400	400
39	300	300	0	1500	50	50	0	250	50	50	0	250
41	300	300	1500	0	50	50	250	0	50	50	250	0

3.4. Experience with VS-PLUS

This subsection is a brief description of our experience with VS-PLUS at California PATH. Our goal was to do an evaluation of the VS-PLUS plug-in for Paramics using the same scenarios and the same test intersection as in Section 3.3.

VS-PLUS is a software suite for the development of fully actuated traffic control systems. VS-PLUS follows its own particular control strategy, but it can be used to conform to the NEMA standard. The implemented control code can be "compiled" into executable code to be run on a 2070 controller or to be run on a software emulated 2070 controller that can communicate with a Paramics model using a provided Plug-in. For more information about VS-PLUS, see Section 4 in this document.

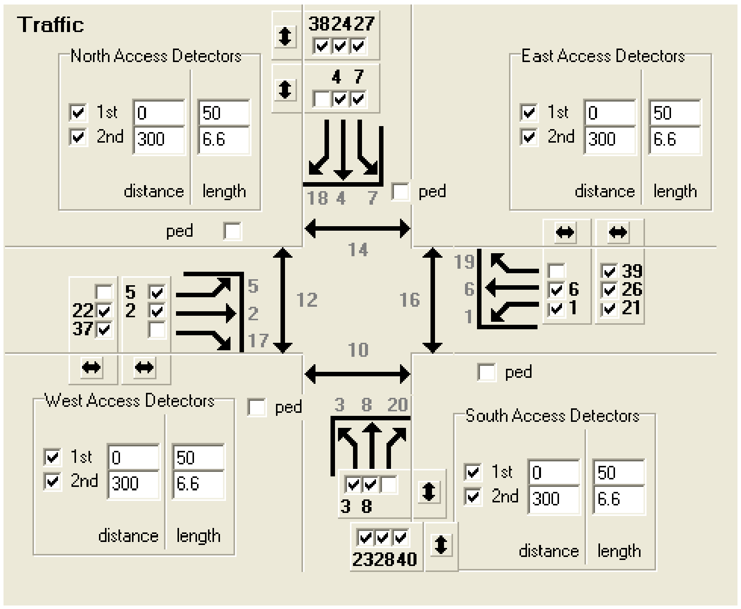
The purpose of the work was to understand the VS-PLUS software suite, restricting attention to the NEMA standard, and to create a traffic controller software simulator for the test intersection. This exercise was aimed at detecting any shortcomings of the VS-PLUS suite and comparing its performance with the real 2070, the plug-in described above in Section 3.3, as well as additional plug-ins being developed at UC Berkeley.

In this subsection, we first describe modifications that had to be made to the Paramics projects because of limitations in the way the VS-PLUS for NEMA standard works. These limitations required modification of the Paramics project for the test intersection. The modifications were minor but they may lead to behavioral differences when compared with the other controllers.

Required modifications to the VS-PLUS plug-in

In order to compare the behavior of the controller software plug-in models with HiLS controller operation, all controllers must be run on the same environment (i.e. same intersection, same detector layout and same parameters). If possible the same Paramics intersection model should be used. Unfortunately the test intersection used for the tests in Section 3.3 cannot be used as is with VS-PLUS for NEMA. As is shown in Figure 3.19, the VS-PLUS NEMA wizard allow two rows of sensors for each approach. All the sensors of each row are required to have the same length and distance from the intersection. The Paramics model of the test intersection, as can be seen in Figure 3.20, features sensors of different length in the same row.

The problem was addressed by modifying the Paramics model to shorten the length of the sensor of the left turn lane to the length of the sensors in the other lanes. A second problem arose with the number of sensors per direction of traffic. VS-PLUS allows a single sensor per approach (see Figure 3.19), while, as can be seen in Figure 3.20, in the Paramics model there is a sensor per lane and multiple lanes for each direction of traffic. The problem cannot be addressed by replacing the sensor with multi-lane sensors (one for each direction of traffic, because Paramics supports detectors that are either a single lane wide or that span across all the lanes. To address the problem at the plug-in level would require access to the VS-PLUS source code for Paramics. An attempt was made to address the problem within VS-PLUS (using VS-PLUS work-suite to modify the project created with the VS-PLUS NEMA wizard).


Traffic

North Access Detectors

	distance	length
☑ 1st	0	50
☑ 2nd	300	6.6

East Access Detectors

	distance	length
☑ 1st	0	50
☑ 2nd	300	6.6

West Access Detectors

	distance	length
☑ 1st	0	50
☑ 2nd	300	6.6

South Access Detectors

	distance	length
☑ 1st	0	50
☑ 2nd	300	6.6

Figure 3.19: VS-PLUS NEMA wizard: all sensors in the same row have same length and distance.

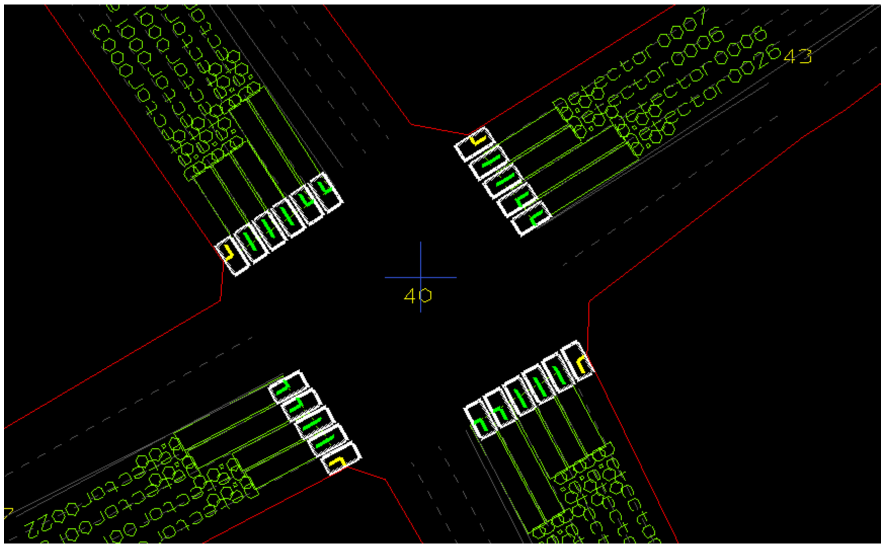


Figure 3.20: Paramics model: stop line detectors different lengths, multiple sensors per direction

VS-PLUS NEMA emulator parameters

The second step was the creation of the software traffic controller using VS-PLUS. In order to do this it was necessary to provide to VS-PLUS some parameters values (e.g. detector positions or recall algorithm to be used). These were retrieved from the Paramics intersection model or from the 2070 controller itself. This subsection shows all the parameters required to set up a NEMA emulator using VS-PLUS. The intersection approaches are labeled with the numeric codes in Figure 3.5. The phases are labeled according to the NEMA standard (1 to 4 for the first ring, 5 to 8 for the second ring, 10, 12, 14 and 16 for pedestrian and 17, 18, 19 and 20 for right turns), as represented in figure 4. The parameters are organized in 7 groups: intersection, phases, overlaps, phasing sequence, permissive periods, detectors and controller generation.

Table 3.21: VS-PLUS NEMA emulator intersection parameters

Parameter	Description	Value given	Source
Main street	Specify the main direction	37 → 43 (W-E)	Intersection geography
Secondary street	Specify the secondary road main direction	39 → 41 (N-S)	Intersection geography
Approaches	Layout description of the intersection. In here it is possible to specify for each road what are the traffic directions supported.	ALL	Aerial photographs of the intersection (see figure 1)
Intersection Name	Name to be given to the controller	Barancal Intersection	Documentation from the HiL project
Version	For version control	2.0	
Options	Specifies if there is emergency preemption	NO	2070

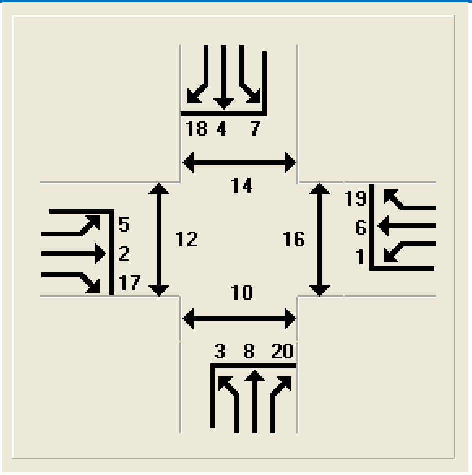


Figure 3.22: standard NEMA convention is used to denote the traffic movements

Table 3.23: VS-PLUS NEMA emulator Phases

Parameter	Description	Value given	Source
Cars	Specifies the vehicle-related phases	ALL active	2070
Peds	Specifies the pedestrian-related phases	NO	2070

Table 3.24: VS-PLUS NEMA emulator Overlaps

Parameter	Description	Value given	Source
17	Right turn	Overlapping with 2	2070
18	Right turn	Overlapping with 4	2070
19	Right turn	Overlapping with 6	2070
20	Right turn	Overlapping with 8	2070

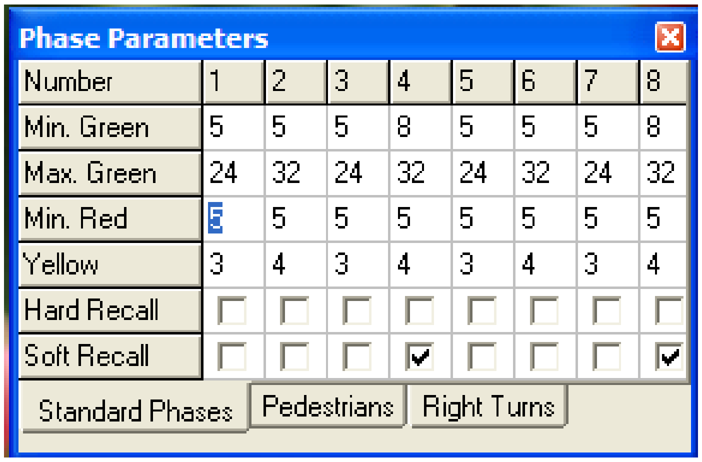
Default 2070 phase sequences (lead-lead left turns) were used during simulation. Please see UC Irvine actuated signal plugin documentation (Liu 2001, Liu 2003) for an overview of turning movements, NEMA phases, rings, and ring phases.

Table 3.25: VS-PLUS NEMA emulator Phasing Sequence

Parameter	Description	Value given	Source
Ring 1 (1 2)	Order between the two phases between a barrier	Lead	See Section 3.3.
Ring 1 (3 4)	Order between the two phases between a barrier	Lead	See Section 3.3.
Ring 2 (5 6)	Order between the two phases between a barrier	Lead	See Section 3.3.
Ring 2 (7 8)	Order between the two phases between a barrier	Lead	See Section 3.3.

Table 3.26: VS-PLUS NEMA emulator Phase Timing

Parameter	Description	Value given	Source
All red	How long the intersection presents a red signal to all directions	1 sec	2070
Show list (phase parameters)	Various details like minimum green, max green, etc	See the screenshot in figure 5	2070


Phase Parameters

Number	1	2	3	4	5	6	7	8
Min. Green	5	5	5	8	5	5	5	8
Max. Green	24	32	24	32	24	32	24	32
Min. Red	5	5	5	5	5	5	5	5
Yellow	3	4	3	4	3	4	3	4
Hard Recall	☐	☐	☐	☐	☐	☐	☐	☐
Soft Recall	☐	☐	☐	✓	☐	☐	☐	✓

Figure 3.27: Phase parameters for the test intersection

The following table represent the mapping between the Paramics detectors and the VS-PLUS detectors, with the first four columns corresponding to the Paramics project file. This reflects the VS-PLUS shortcomings discussed previously:

- VS-PLUS support only 2 sensors per direction of traffic per access. This is not good if a direction of traffic has multiple lanes!
- VS-PLUS requires the length of all the sensors in a row to have the same length. This is not always the case!

Table 3.25: VS-PLUS NEMA emulator Phasing Detectors

Number	Distance	Link	Lane	Length	VS-PLUS
1	0	39:40	3	50	**4**
1a	0	39:40	2	50	**4**
2	300	39:40 upstream	1	6.6	**24**
2a	300	39:40 upstream	2	6.6	**27**
3	0	39:40	1	50	**4**
23	0	39:40	4	70	**7**
24	0	39:40	5	70	**7**
25	300	39:40 upstream	0	6.6	**38**
4	300	43:40 upstream	2	6.6	**39**
5	300	43:40 upstream	1	6.6	**39**
6	50	43:40	2	50	**6**
7	50	43:40	1	50	**6**
8	70	43:40	3	70	**1**
26	70	43:40	4	70	**1**
29	300	43:40 upstream	0	6.6	**39**
9	50	41:40	1	50	**8**
10	50	41:40	2	50	**8**
11	300	41:40 upstream	1	6.6	**28**
12	300	41:40 upstream	2	6.6	**23**
13	50	41:40	3	50	**8**
19	70	41:40	4	70	**3**
20	70	41:40	5	70	**3**
21	300	41:40 upstream	0	6.6	**40**
14	50	37:40	2	50	**2**
15	50	37:40	1	50	**2**
16	70	37:40	3	70	**5**
17	6.6	37:40 upstream	0	6.6	**37**
18	6.6	37:40 upstream	1	6.6	**22**
22	70	37:40	4	70	**5**

Parameter	Description	Value given	Source
1st distance	From stopbar (ft)	0	**Paramics project**
1st length	(ft)	50	**Paramics project**
2nd distance	From stopbar (ft)	300	**Paramics project**
2nd length	(ft)	6.6	**Paramics project**

Performance of VS-PLUS

After performing all the above modifications, an attempt was made to use VS-PLUS to generate the two “vcb” files that are needed by the VS-PLUS 2070 emulator. Unfortunately, while one can be generated easily, the generation of the second does not even start from the NEMA wizard. We contacted VS-PLUS to have the issue addressed, sending them the project file. VS-PLUS assess that our file was correctly written, duplicated the problem using the same software release we are using, and said they would send us a patched version to fix the problem. Unfortunately the patch did not arrive in time for us to do the simulations matched to the previous HiLS runs.

While waiting for the software patch it was not possible to test the performance of VS-PLUS on the test intersection described in Section 3.3. We did carry out some simulations on the test intersections provided with VS-PLUS. One of these standard intersections involved 8 simulated traffic controllers. The Paramics simulation with this system was conducted with a 1.8Ghz, 512 MB of RAM system running Windows XP Professional. The Paramics simulation took 2 minutes to initialize (slow initialization), then another 30 seconds to start the simulation (slow start), but then was extremely fast to run , taking only 15 minutes to simulate 1 hour of traffic for eight traffic controllers.

3.5. Conclusions and Further Work

The results obtained so far indicate that the HiLS using a CID with 170 and 2070 controllers is valuable primarily for two purposes:

- Testing of new traffic signal controller logic for correctness in a simulation environment in advance of field testing.
- Validation of software traffic signal control modules to be used in microscopic simulations of intersection networks.

The latter purpose might not be required if on-controller software were integrated with the software emulation model, as it is for VS-PLUS. With Caltrans TSCP and with the vendor software provided by other companies supporting the 170 and 2070, such integration does not seem to be available. Even with VS-PLUS, although the performance of the software emulator seems reasonable, a more efficient though less detailed model might be desirable to determine performance in large networks of intersections.

In the HiLS work done so far, there are many improvements that could be made:

- Currently each HiL simulation is manually started. It appears the Paramics API has some functions that allow the network to reload and use all plug-ins. Proper use of these functions can allow the user to set the number of simulation runs desired. This will allow simulation runs to iterate automatically up to a specified number. The auto-iterating capability will remove the need for the user to be physically present at the test platform to manually reload simulations.
- The HiL plug-in has hard coded values within the source code, so if one uses the code for another intersection, or a completely different network, all of these values need to be changed to reflect the intersection or network used and the code needs to be recompiled. Since Paramics allows the writing and reading of configuration files, the

code in the plug-in can easily be modified to set these key values to values read from the configuration file. A small program can be written to build a configuration file using user values. This will allow easy manipulation of network dependent variables which include: intersection links, controlled node, link detectors, CID channel timing, and desired time-step. Once this is done, the plug-in will be more portable and able to be used for HiL verification in a variety of simulations.

- Time did not allow the same network simulations to be run with a 170 controller. The 2070 should be substituted with a 170 controller, to see if there are measurable difference compared to each other and compared to the software plug-in.
- A final task involves simulation of a corridor. Henry Liu has tested a corridor using configuration files and the *UC Irvine Actuated Signal Control plug-in*. It would be beneficial to verify these results with a full HiL simulation (hardware control of all intersections) or even with a partial HiL simulation (hardware control of some intersections) to compare traffic performance of the corridor between software emulation and hardware control.
- Comparison between HiLS and VS-PLUS emulation would also prove beneficial, both for clarifying any essential performance differences between the VS-PLUS NEMA set-up and the typical set-up of 170 and 2070 control logic, and for checking the accuracy of the VS-PLUS emulator against a 2070 with the VS-PLUS software loaded.

There are other general issues with the Paramics simulation environment that are outside the scope of this report. The Paramics application is not straightforward and the documentation sometimes contains typos or dead links to sections. The accuracy of the simulation is also difficult to control. For example, the simulated network had problems with driver lane choices. Within Paramics, one can configure the driver lane choices based on origin and destination. After reconfiguring the lane choices, there were still issues with the Paramics driver model. In addition, Paramics does not reference detector location to the stopbar or intersection node. One must manually look over the detector configuration file and note distances from kerb points. Although driver lane choices decrease simulation accuracy, these issues were observed in both HiL and actuated signal simulations. Therefore, the issues do not prevent results from being compared.

4. User's guide: Integrated Paramics/VS-PLUS Simulation

4.1 Introduction to VS-PLUS

4.1.1 VS-PLUS

VS-PLUS is a stand-alone signal control program that was developed specifically for traffic responsive signal control systems .Since its first implementation in Switzerland in 1983, VS-PLUS has been installed at more than 1000 intersections throughout Europe. In the United States, the first implementation was in Vancouver, WA in 2001. The VS-PLUS software provides a user-friendly visual interface for developing and evaluating signal control parameters. It can time fully actuated and coordinated traffic signal control systems and handle complicated intersections as six-leg and diamond interchanges. VS-PLUS interface software packages were developed recently for interfacing with microscopic traffic simulation models. By interfacing with microscopic simulation models, VS-PLUS overrides their built-in control functions and serves as a virtual controller in simulation.

VS-PLUS is a parameter-driven program. The control functions are carried out through the definition of various signal timing parameters in the program. Signal timing parameters are generated on the basis of two principles, i.e., "optimum" and "plausible". The objective of the "optimum" process is to minimize total intersection delay, while the purpose of the "plausible" process is to compromise the waiting times of the vehicles on each individual intersection approach. That is, motorists at a particular direction should not be kept waiting longer than those on the other directions for no apparent reasons. Note that the two objectives conflict with each other when traffic are heavy and unevenly distributed among intersection approaches. Although application guidelines for both approaches are provided within in the report, we would suggest use the "optimum" approach for all traffic conditions.

VS-NEMA Wizard

The VS-NEMA Wizard is a supplemental tool provided with VS-PLUS. It was developed specifically for American users for the timing design of NEMA controllers. In current version, it can handle standard four-leg intersections with signals for pedestrian crossing and emergency vehicle preemption. It takes seven steps for timing a standard NEMA controller in VS-NEMA. The results of the optimized signal parameters can be downloaded directly to the controllers running VS-PLUS. Note that VS-PLUS itself is a control program that generates signal timing parameters, VS-NEMA Wizard has most of the features of VS-PLUS and the role it plays is to restrict the designing process and the signal parameters generated follows the NEMA standard.

VS-WorkSuite

VS-WorkSuite (Verkehrs Systeme 2006) is an object-oriented planning and development user interface for VS-PLUS. It is oten used for generating signal timing plans from VS-PLUS. It can also be used for evaluating the signal parameters generated by other signal control programs. VS-WorkSuite provides open XML file interfaces for exchanging data with other signal control programs. For example, signal timing parameters as clearance time matrices, phase groups, offset time matrices and so forth can be imported to VS-PLUS for evaluation through the XML interfaces.

Through the interface with microscopic simulation, a VS-PLUS signal timing plan can be evaluated in a simulation model and the signal timing parameters can be modified during simulation tests. The VS-WorkSuite also provides an on-line Monitor that shows main control parameters on the screen to allow for the users to track the performance of the signal timing plan being tested..

4.1.2 Terminology

VS-PLUS uses unique terminologies to define its signal control parameters, which are not identical to those used by closed-loop traffic signal control systems. Some of the critical terminologies used by VS-PLUS is interpreted in the following. Understanding of these terms are of great importance to manipulating VS-PLUS as well as understanding its control philosophy. .

Display Element

The display element refers to all possible types of displays that can be controlled by a controller with an input/output (I/O) instruction. A display element typically corresponds to a specific single signal group; however, it can also correspond to other possible signal signs, which include:

- Pedestrian signal associated with pedestrian push buttons,
- Variable message sign, and
- LED of the controller operating panel.

Phase

A phase in VS-PLUS is defined as a control loop that consists of various monitoring elements (mainly detectors) and a display element. A phase type can be of any of the individual elements as shown below or a combination of them:

- Passenger cars,
- Public transit (PT),
- Pedestrians (Ped),
- Bicycles and bicyclist and/or
- Emergency vehicles (EV).

Accordingly, it is also possible for a phase to have more than one display element and meanwhile, one display element can be controlled by more than one phases. For example, the passenger car phase and PT phase can correspond to a same display element.

Permissive Signal (frame signal)

The concept of the permissive signal is critical in VS-PLUS. It is important to note that the permissive signal in VS-PLUS does not mean a protected/permissive signal phase used in common signal control systems. In VS-PLUS, the permissive signals are the control signal or frame signal that corresponds to each phase. They serve as coordinating instruments. The permissive signal is defined in the form of a call range and an extension range. The call range is the time period within a signal cycle when a phase can be called by detectors or by other associated phases. The extension range is the time period within the cycle during which the green time of a current phase can be extended.

Permissive Period Plan (frame signal plan)

The sum of all permissive signals for all phases generates what is called a permissive period plan. The permissive period plan is initially generated on the basis of a signal timing plan, which is a fixed signal timing plan for actuated signals. As in a closed-loop system, controllers run several different timing plans during the time of day. Such timing plans are also prerequisite for designing the permissive period plan in VS-PLUS. In a closed-loop system, the permissive period is normally used as a control command to switch signal phases. In VS-PLUS a permissive period plan is used to develop control strategies . For example, Figure 4.1 shows several types of permissive period plans. Details about the permissive period plan can be found in VS-PLUS ‘s user manual. The guidelines with regard to how to use the permissive plan are introduced in detail with several application examples in this report.

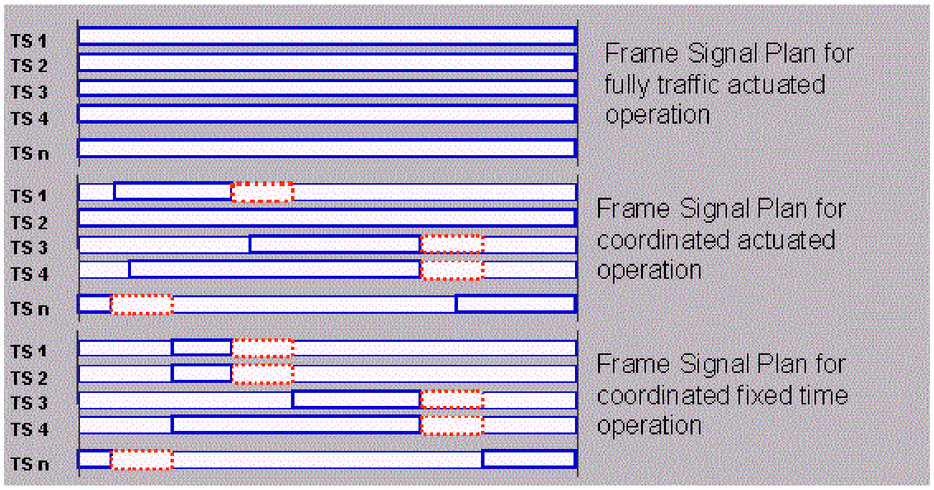

TS 1
TS 2
TS 3
TS 4
TS n
Frame Signal Plan for fully traffic actuated operation
TS 1
TS 2
TS 3
TS 4
TS n
Frame Signal Plan for coordinated actuated operation
TS 1
TS 2
TS 3
TS 4
TS n
Frame Signal Plan for coordinated fixed time operation

Figure 4.1 Permissive Period Plans

4.1.3 The control philosophy of VS-PLUS

A traffic responsive signal control system relies on the coordinated efforts between traffic monitoring devices and traffic signal controllers. In VS-PLUS, all detector calls must be analyzed through the use of the detector evaluation. In order to accomplish this, a control parameter, detector waiting time (TWDET), is assigned to each detector. The detector waiting time is started with the first detector call and is reset when its corresponding phase receives green. In the process of detector evaluation, VS-PLUS checks the detector waiting time for each detector at every millisecond.

Phase evaluation is then processed based on the results of detector evaluation. During phase evaluation, VS-PLUS checks each phase to determine whether the phase is to be called or is in green extension time. Another control parameter, phase waiting time (TWVS), is assigned to each phase at this time. The phase waiting time is measured from the time when the first detector call was trigged. A detector call only results in a phase call if the permissive signal of the corresponding phase is within its call range and the phase waiting time has reached its maximum value. The maximum waiting time can also be defined for each phase.

The result of the phase evaluation process is a status value for each phase. The priority value can then be calculated based on these status values. The priority value provides the basic information for the process called “picture development”, which is the core of VS-PLUS model.

The picture development process determines main and corresponding minor phases of the target intersection. The result of the picture development process is a target picture. This target picture lists all phases in a descending order that are next in line to receive green time and that are not conflicting with the higher priority phases.

The target picture is then processed in the switching section. An “on” or “off” command is assigned to each phase at this stage. All phases in conflict with the target picture are assigned an “off” command. The “on” and “off” commands for the phases are then converted into the corresponding display elements in the VS-PLUS interface. VS-PLUS then calculates the phase change on the basis of the phase change time matrix and the offset. The whole process takes place every second and the function of signal control is then carried out by the control parameters, which are designed by the design wizard.

Detector Evaluation

Detector evaluation is the first step in the process of VS-PLUS. In this step, the detector waiting time will be counted for each detector, starting with the first detector’s call. The detector call is processed differently depending on the detector type used. The two types of detectors used in VS-PLUS are impulse detectors and occupancy detectors. The occupancy time parameter is decisive for determining the type of detector. If the occupancy time of a detector is set to zero, the detector is an impulse detector. Otherwise, the detector is an occupancy detector.

<u>Impulse Detector</u>

The detector waiting time is started upon the passage of the first vehicle. In order to avoid an incorrect call, a hold time needs to be set for each detector. If the net gap for a detector is greater than the hold time, the detector waiting time will be stopped and reset.

<u>Occupancy Detector</u>

The detector waiting time is started when the occupancy time of a detector is greater than its set value. If the detector is no longer occupied after the detector waiting time has started, the detector waiting time will be reset.

Phase Evaluation

The detector waiting time for each detector is obtained from the detector evaluation. In the phase evaluation process, all phases are evaluated, and a timer for the phase waiting time is then assigned for each phase. Phase waiting time starts with the first corresponding detector call. If the detector waiting time is reset, the phase waiting time is also reset.

<u>Phase Call</u>

A detector call will activate the corresponding phase call. Two conditions must be met in order to call a phase. They are:

- the detector waiting time is greater than the delay time, and
- the corresponding permissive signal must be in the call range.

If the detector call is outside of the call range of the permissive signal, the phase will be called at the start of the next permissive signal. The phase can also be called by the green flag or when the phase waiting time exceeds the maximum phase waiting time.

<u>Phase Status</u>

The phase evaluation process checks the status of each single phase.

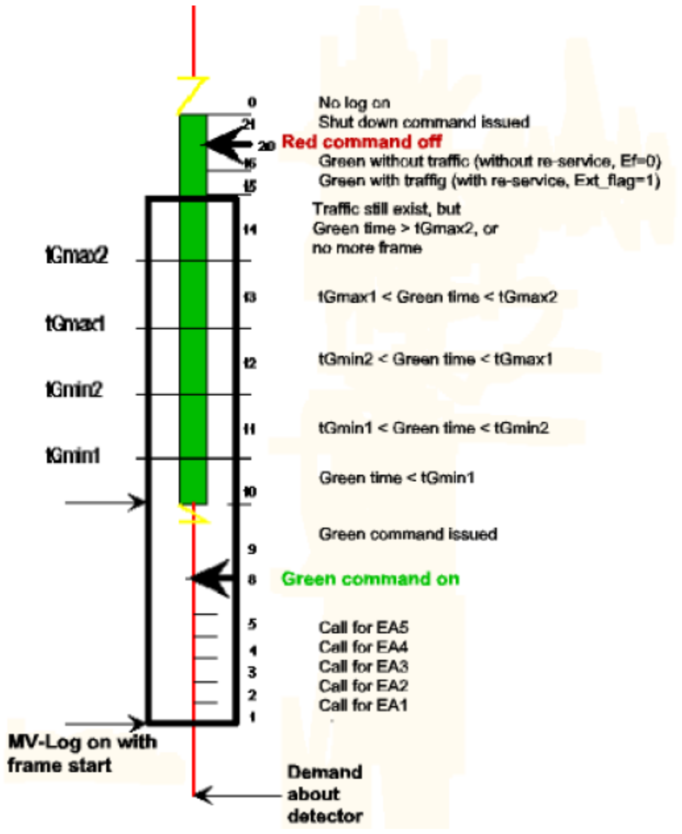


Figure 4.2 Phase status values (1)

As shown in Figure 4.2, the status of a phase is summarized by its status values:

Status value	**Description**
The phase is red	
0	The phase is inactive.
1	The phase has the intervention level 1

2	The phase has the intervention level 2
3	The phase has the intervention level 3
4	The phase has the intervention level 4
5	The phase has the intervention level 5
8	The phase can be started
9	The phase was set to green

The phase is green

10	Green time < tGmin1
11	tGmin1<Green time<tGmin2
12	tGmin2<Green time<tGmax1
13	tGmax1<Green time<tGmax2
14	Green time>tGmax2 and traffic still exist
15	No traffic, can be extended by extension flag
16	No traffic, can not be extended
19	Special red command for the phase
20	The phase can be set to red
21	The phase was set to red.

A phase status can be classified in two groups—phase in red and phase in green.

Phase in Red

When the phase is in red, the phase will be evaluated when the phase waiting time is greater than zero and its permissive signal is in its call range, or the maximum phase waiting time is reached. The intervention level of the evaluated phase determines the green time of its conflicting phase. Higher intervention levels lead to a shorter green time of the conflicting phase. If the phase waiting time exceeds the control time, the intervention level will also be raised. As shown in Figure 4.3 VS-PLUS recognizes five different intervention levels.

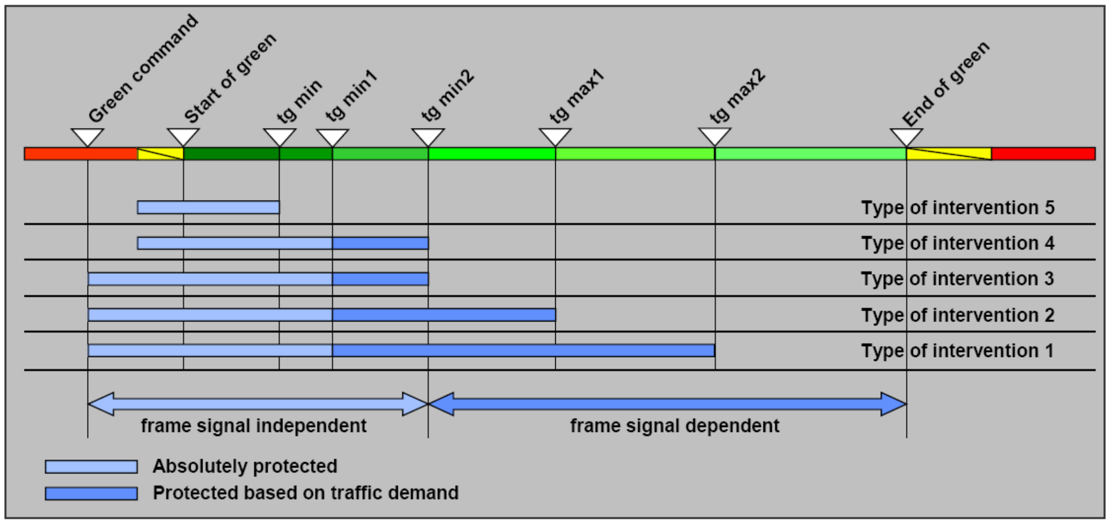


Figure 4.3: Intervention Level (4)

The weakest type of intervention (level 1) represents the time range starting from the beginning of the green command to the end of the maximum possible extension of green. It is the most common type of intervention that initiates a new phase and would run the normal phase sequence if there are no higher priority interventions. The strongest and most extreme type of intervention (level 5) merely respects the minimum green time safeguarded by the controller. High level interventions may intervene lower level interventions by cutting their corresponding green intervals shorter than designed. A green command can also be totally withdrawn after having been issued. This type of intervention is used in exceptional cases only (i.e., emergency vehicles or railroad preemption).

Phase in Green

If the phase is in green, it is examined to determine whether or not the phase needs to be extended. The green time of the phase is restricted on five levels corresponding to the intervention levels of its conflicting phase.

- Green time< min green time 1 (tGmin1)
- min green time 1 (tGmin1)<Green time< min green time 2 (tGmin2)
- min green time 2 (tGmin2)<Green time< max green time 1 (tGmax1)
- max green time 1 (tGmax1)<Green time< max green time 2 (tGmax2)
- Green time> max green time 2 (tGmax2)

Picture Development

Only those called phases are evaluated in picture development. In this step, the phases that will be served are determined based on their called phase status and priority values. The picture development process works with the priority element (PE) whose working scheme is shown by Figure 4.4. The priority elements range from 1 to 6 and follow the basic structure as shown in Figure 4.4.

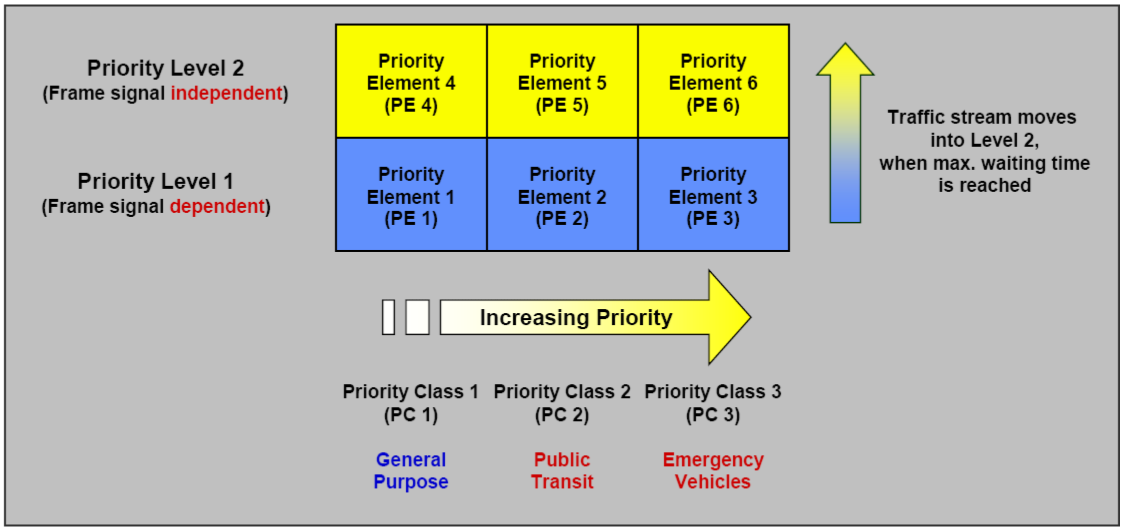

Priority Level 2
(Frame signal independent)
Priority Level 1
(Frame signal dependent)
Priority Element 4 (PE 4)
Priority Element 5 (PE 5)
Priority Element 6 (PE 6)
Priority Element 1 (PE 1)
Priority Element 2 (PE 2)
Priority Element 3 (PE 3)
Traffic stream moves into Level 2, when max. waiting time is reached
Increasing Priority
Priority Class 1 (PC 1)
Priority Class 2 (PC 2)
Priority Class 3 (PC 3)
General Purpose
Public Transit
Emergency Vehicles

Figure 4.4 Priority Structure in VS-PLUS (4)

There are three priority classes and two priority levels in the priority structure. The priority classes are:

- Priority Class 1: normal sequence (general-purpose traffic, cyclists and pedestrians),
- Priority Class 2: higher priority (public transport) and
- Priority Class 3: special interventions (emergency, heavy rail, etc.).

Priority elements 1, 2, and 3 belong to priority level 1, while priority elements 4, 5 and 6 correspond to priority level 2. The classification of two priority levels is related to the phase waiting time. For example, a phase with a priority element 1 will be upgraded to priority element 4 if its phase waiting time exceeds the maximum waiting time.

The first step of the picture development process determines the main phase. A pointer is present in each priority element that points to the phase that can be selected as the main phase and defines the sequence of the main phase. There are a maximum of six main phases in VS-PLUS because there are only six priority elements. Since a main phase sequence is present for each priority element, the same main phase sequence is used for levels 1 and 2 of the priority classes. The evaluation procedure is identical for every priority element.

The second step of the picture development process determines the minor phases. A minor phase sequence is determined for each main phase. VS-PLUS provides two types of minor phase sequences:

- minor phases with calls. This type of minor phases will be considered only if they are being called. And,
- minor phases without calls. This type of minor phases will be considered even if there are not calls on it.

Switching

The result of the picture development is the target picture, which is a list of phases. The target picture lists all phases that will be switched to green, and their conflicting phases will be switched to red. If a phase is to be switched to green, it must be examined to determine whether those minor phases that have been called are to be included. In addition, when a phase is on green, it must be examined to determine whether it has a “red flag”. A “red flag” is a command through which a user can intervene normal operational processes and defines a specific phase to red at a specific time (normally, a phase remains on green until it is switched to red by its conflicting phase).

Interface

The interface assigns phases to their corresponding display elements and issues “on” and “off” commands to them. VS-PLUS then examines the time for each phase change and the commands for switching will be sent to the signal groups.

4.2 Evaluation of Paramics/VS-PLUS Interface

4.2.1 Operating integrated Paramics/VS-PLUS software

Paramics is a microscopic traffic simulation model which is being used extensively by Caltrans engineers in their planning, design, and control projects. The Paramics Project Suite consists of Modeler, Processor, Analyzer, and Programmer. Details about the Paramics model can be found in Paramics User’s manual and will not be described in this report.

VS-PLUS is interfaced with Paramics through a plug-in file. In order to use the plug-in in Paramics, it needs to be installed and follows the following instructions for registration.

Installation

(1) Move three files 'ZIPDLL.DLL', 'UNZDLL.DLL' and 'vspLIcense.dll' into user’s 'ParamicsV5' directory.

(2) Move the directory 'VS-PLUS' into user’s 'ParamicsV5/plugins' directory.

Registration

(1) Launch 'Modeler' and open the Paramics model with VS-PLUS signal setting.

(2) A message will appear inside the text area for registration: 'VS-PLUS Please Register!'.

(3) Exit from 'Modeler' and move to the 'ParamicsV5/plugins/VS-PLUS' directory.

(4) Mail the file 'toBeLicensed.cfg' to 'Thomas.Riedel@VS-PLUS.com'.

(5) When you receive the license file, copy it into the 'ParamicsV5/plugins/VS-PLUS' directory.

(Note: The application 'Paramics Viewer' is not yet supported by this plug-in.)

Figure 4.5 depicts the logical relationship between Paramics and VS-PLUS in the integrated model. VS-PLUS functions as a virtual controller in Paramics by overriding the built-in signal control functions of Paramics. The signal control parameters are generated in VS-PLUS through the processes of Controller Setting, Parameters Setting, and creation of Parameter files. Controller setting is a primary process in VS-PLUS that defines the basic topology of the signal control functions; Parameter setting is to define control logics of VS-PLUS signals. The signal control parameters generated by VS-PLUS are then converted to Paramics parameters in the process of creating Paramics files.

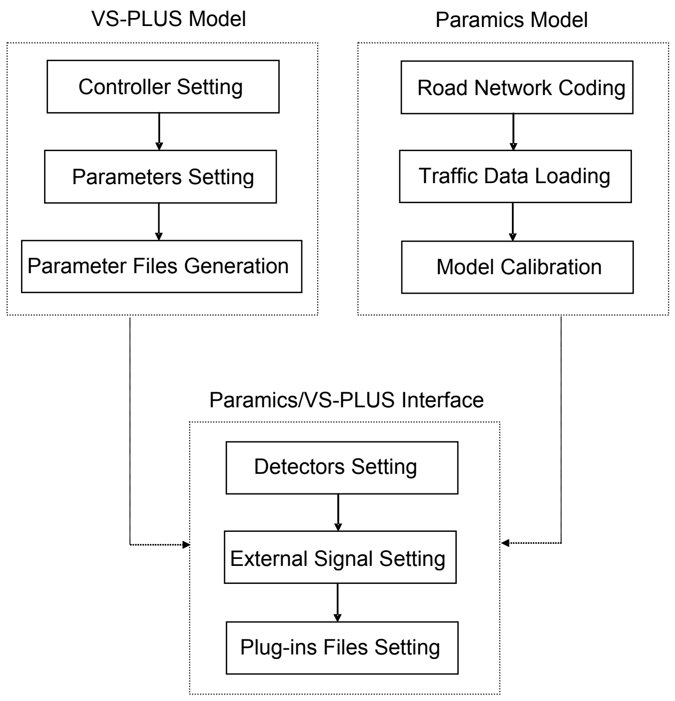


Figure 4.5 Logical relationship between Paramics and VS-PLUS

Once the signal control parameters are converted, three major procedures, namely, detector setting, external signal setting, and plug-ins files setting need to be defined in Paramics/VS-PLUS so that VS-PLUS controllers can work in the integrated environment. In the following, we use three examples to illustrate in detail how to operate the integrated Paramics/VS-PLUS simulation model. The first example is application of Paramics/VS-PLUS to a diamond interchange, through which, we aim to introduce its capability as well as necessary procedures for timing irregular intersections; the second example demonstrates the model's capability in emulating isolated traffic responsive signal controllers; the third example creates a small network composed of three intersections with actuated and coordinated controllers for the test of the model's capability in coordination.

4.2.2 Diamond Interchange

Diamond interchange is an irregular type of road junction where a section of freeway intersects with an urban or rural highway. It is grade-separated but controlled by one signal controller. The diamond interchange used in the example is located at the intersection of Slide Road and South Loop 289 in Lubbock, Texas. The aerial photo below (Figure 4.6) shows the geometry of the interchange. The signal runs fixed-time plans and the cycle length is 130 seconds during afternoon peak. Table 4.7 gives its signal timing sheet.

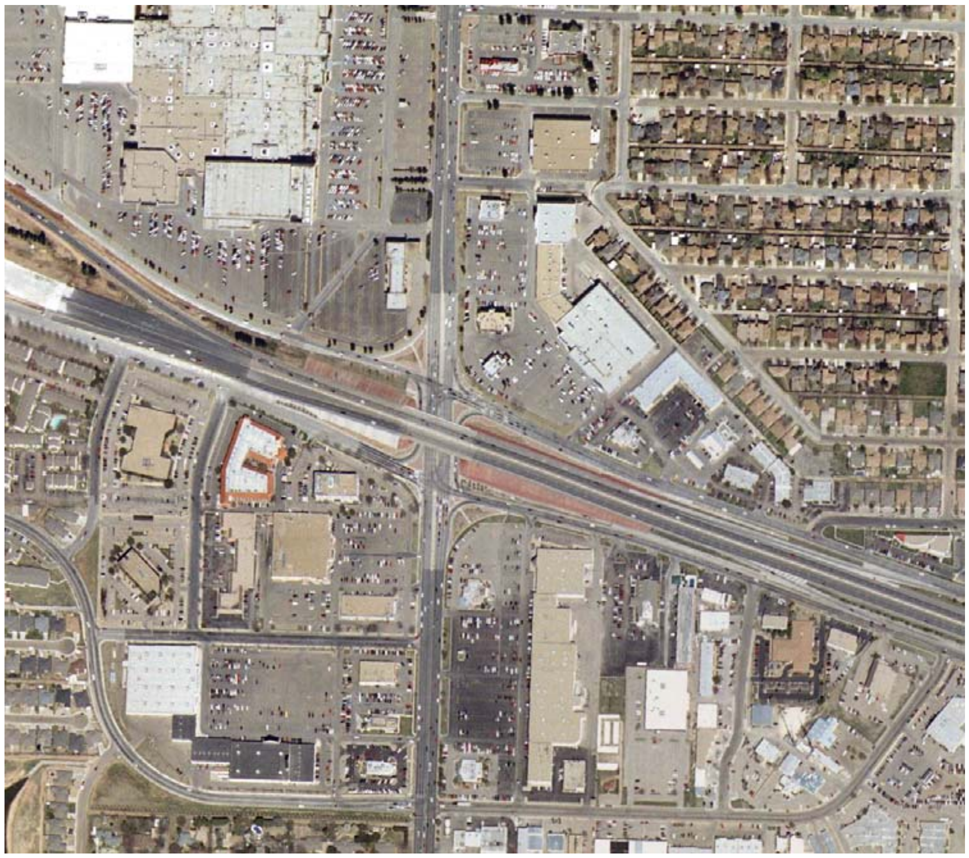


Figure 4.6 Aerial photo of the diamond Interchange

(Source: http://maps2005.ci.lubbock.tx.us/photos.shtml)

Table 4.7 Signal Timing Sheet

Approach	SB1	WB	NB1	EB	SBLT	SB	NBLT	NB	WBLT	EBLT
Phase	Ph1	Ph2	Ph3	Ph4	Ph5	Ph6	Ph7	Ph8	Ph9	Ph10
Min Green	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0
Max Green	41	31	30	28	60	68	46	54	31	28
Min Red	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
Yellow	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Red clear	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0

The signal phases and corresponding traffic approaches are shown in Figure 4.8.

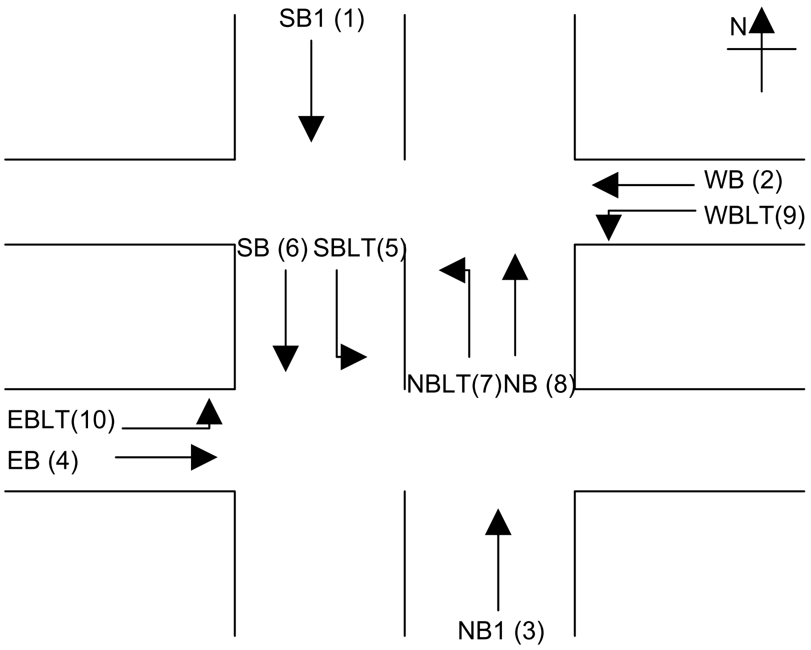
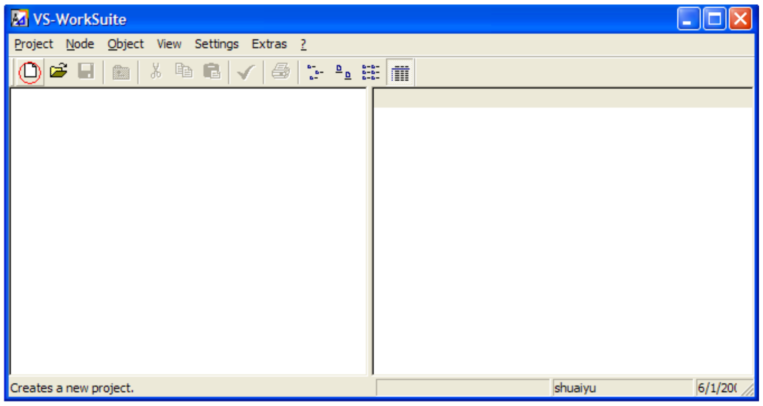


Figure 4.8 Phases of the diamond interchange

VS-PLUS Model

The necessary procedures in building the VS-PLUS model and generating signal parameters in VS-PLUS is introduced step-by-step below.

(1) Create a new VS-PLUS Project



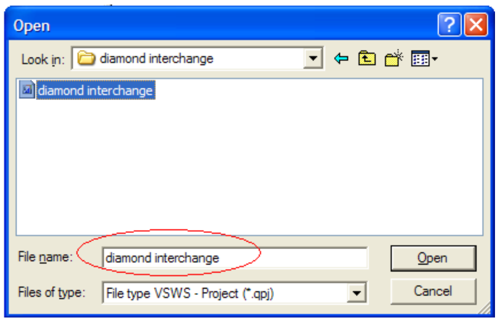
Creates a new project.

shuaiyu

6/1/20(

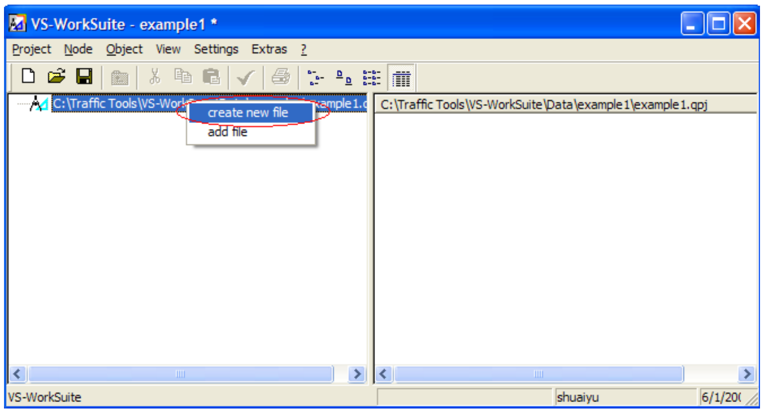
Screen 4.1. Input a project name

Input a project name.


Open
Look in: diamond interchange
diamond interchange
File name: diamond interchange
Files of type: File type VSWS - Project (*.qpj)
Open
Cancel

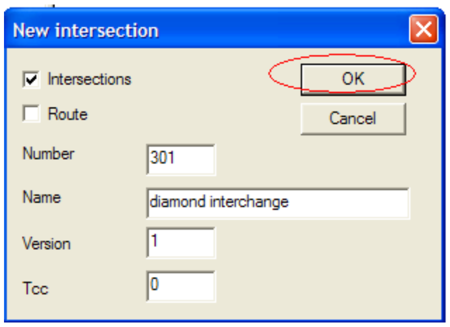
Screen 4.2. Input a project name

Right Click project name to create a new file.


VS-WorkSuite - example1 *
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-Work
create new file
add file
C:\Traffic Tools\VS-WorkSuite\Data\example1\example1.qpj
VS-WorkSuite
shuaiyu
6/1/20

Screen 4.3. Create a new file

Create a new intersection, input name and click ok.

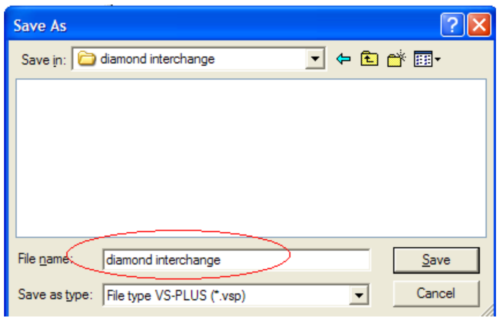


- [✓] Intersections
- [ ] Route

Number	301
Name	diamond interchange
Version	1
Tcc	0

Screen 4.4. Input intersection information

Save the intersection data as *.vsp file (All VS-PLUS parameters will be included in this file).


Save As

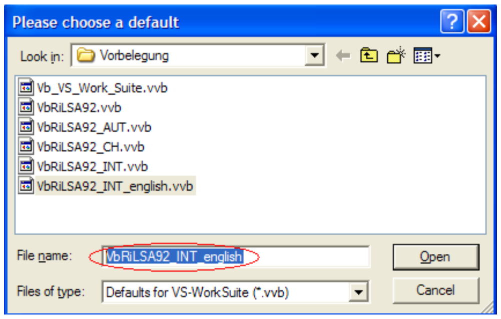
Save in: diamond interchange

File name: diamond interchange

Save as type: File type VS-PLUS (*.vsp)

Screen 4.5. Save the VSP file

Next, choose a default parameter guideline.


Please choose a default

Look in: Vorbelegung

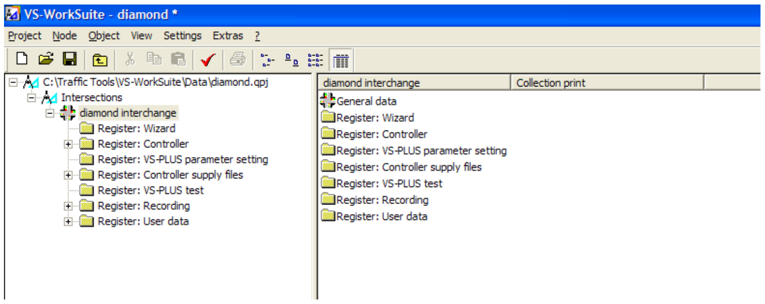
Vb_VS_Work_Suite.vvb
VbRiLSA92.vvb
VbRiLSA92_AUT.vvb
VbRiLSA92_CH.vvb
VbRiLSA92_INT.vvb
VbRiLSA92_INT_english.vvb

File name: VbRiLSA92_INT_english

Files of type: Defaults for VS-WorkSuite (*.vvb)

Screen 4.6. Choose the guideline file

The resulting file is a complete parameter tree.

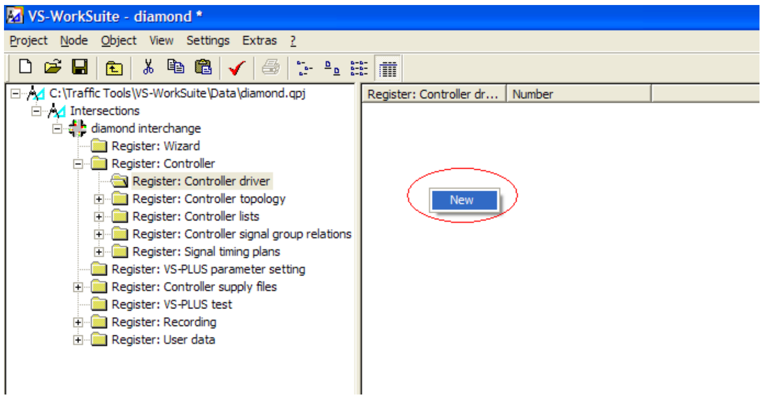

VS-WorkSuite - diamond *

C:\Traffic Tools\VS-WorkSuite\Data\diamond.qpj
Intersections
diamond interchange
Register: Wizard
Register: Controller
Register: VS-PLUS parameter setting
Register: Controller supply files
Register: VS-PLUS test
Register: Recording
Register: User data

diamond interchange	Collection print
General data	
Register: Wizard	
Register: Controller	
Register: VS-PLUS parameter setting	
Register: Controller supply files	
Register: VS-PLUS test	
Register: Recording	
Register: User data	

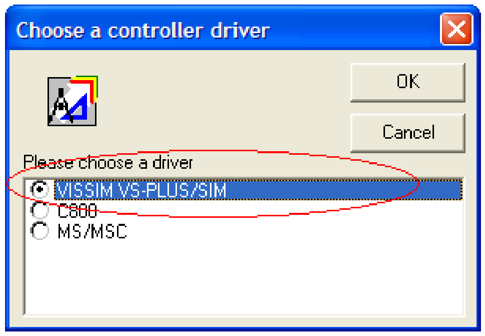
Screen 4.7. The parameter tree

Define a controller driver. Right-click on the list area, and select “new”.


VS-WorkSuite - diamond *
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-WorkSuite\Data\diamond.qpj
Intersections
diamond interchange
Register: Wizard
Register: Controller
Register: Controller driver
Register: Controller topology
Register: Controller lists
Register: Controller signal group relations
Register: Signal timing plans
Register: VS-PLUS parameter setting
Register: Controller supply files
Register: VS-PLUS test
Register: Recording
Register: User data
Register: Controller dr...
Number
New

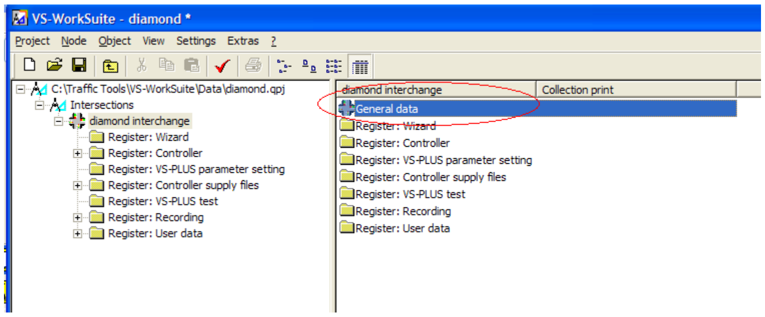
Screen 4.8.	**Create a new controller driver**

Choose a controller driver, such as VISSIM VS-PLUS/SIM.



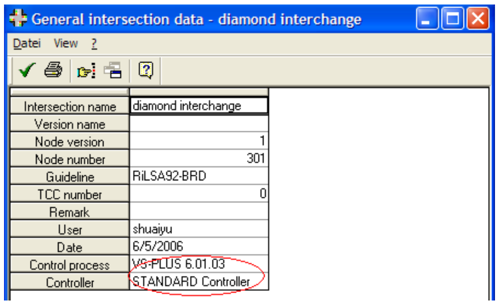
Screen 4.9.	**Choose the type of the controller driver**

Set additional information for the intersection by double-clicking “General Data”.



Screen 4.10.	**Choose general data for the intersection**

Choose guideline, controller process, and controller.

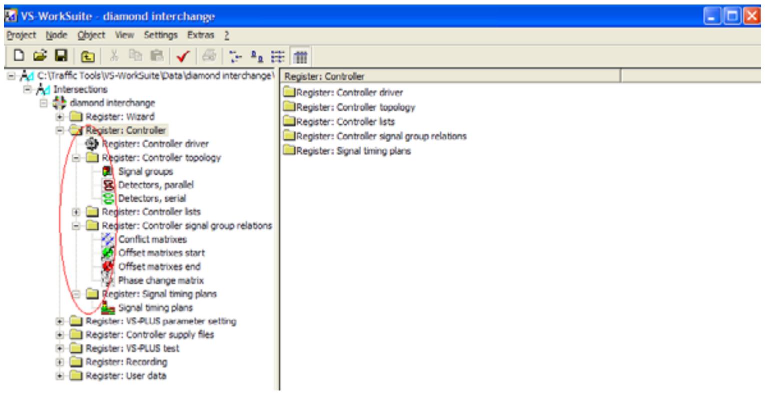


Intersection name	diamond interchange
Version name	
Node version	1
Node number	301
Guideline	RiLSA92-BRD
TCC number	0
Remark	
User	shuaiyu
Date	6/5/2006
Control process	VS-PLUS 6.01.03
Controller	STANDARD Controller

Screen 4.11. Setting the general data for the intersection

(2) Controller Setting

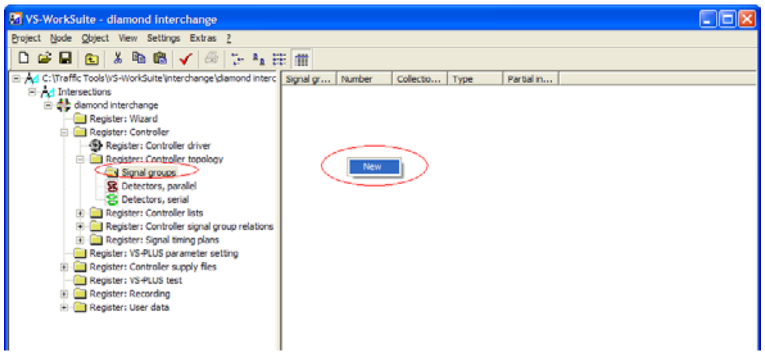
The controller is used to define the basic topology of the signal control. It mainly includes four components, namely, signal groups, detectors, conflict matrixes, and signal timing plans.


VS-WorkSuite - diamond interchange
Project
Node
Object
View
Settings
Extras
?
C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\
Intersections
diamond interchange
Register: Wizard
Register: Controller
Register: Controller driver
Register: Controller topology
Signal groups
Detectors, parallel
Detectors, serial
Register: Controller lists
Register: Controller signal group relations
Conflict matrixes
Offset matrixes start
Offset matrixes end
Phase change matrix
Register: Signal timing plans
Signal timing plans
Register: VS-PLUS parameter setting
Register: Controller supply files
Register: VS-PLUS test
Register: Recording
Register: User data
Register: Controller
Register: Controller driver
Register: Controller topology
Register: Controller lists
Register: Controller signal group relations
Register: Signal timing plans

Screen 4.12. Parameters in the controller group

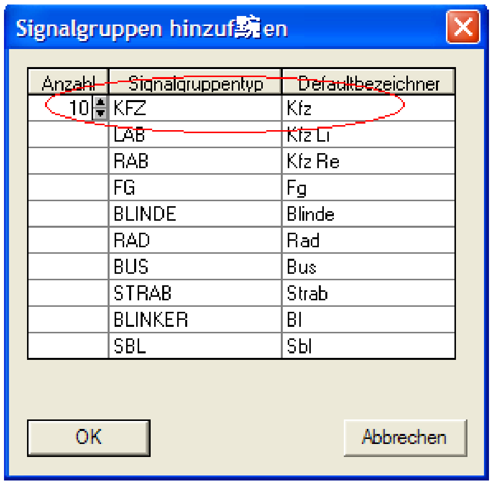
(a) Signal Groups

Define signal groups (right-click after selected signal group on the left list).


VS-WorkSuite - diamond interchange
Project
Node
Object
View
Settings
Extras
?
C:\Traffic Tools\VS-WorkSuite\interchange\diamond interc
Intersections
diamond interchange
Register: Wizard
Register: Controller
Register: Controller driver
Register: Controller topology
Signal groups
Detectors, parallel
Detectors, serial
Register: Controller lists
Register: Controller signal group relations
Register: Signal timing plans
Register: VS-PLUS parameter setting
Register: Controller supply files
Register: VS-PLUS test
Register: Recording
Register: User data
Signal gr...
Number
Collectio...
Type
Partial in...
New

Screen 4.13. Define signal groups

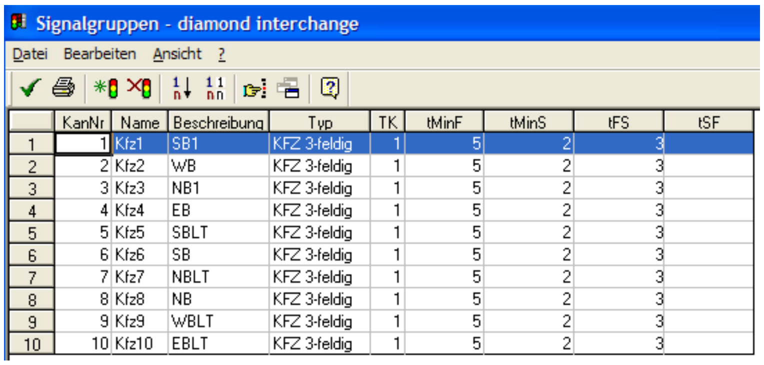
Language translation problems exist in the current version of the software. The following figure shows the signal groups type. For this example, choose 10 "car" signal groups.


Signalgruppen hinzufügen

Anzahl	Signalgruppentyp	Defaultbezeichner
10	KFZ	Kfz
	LAB	Kfz Li
	RAB	Kfz Re
	FG	Fg
	BLINDE	Blinde
	RAD	Rad
	BUS	Bus
	STRAB	Strab
	BLINKER	Bl
	SBL	Sbl

Screen 4.14. Select signal groups

Define each signal group minimum green time as 5.0 seconds, minimum red time as 2.0 seconds and yellow time as 3.0 seconds.


Signalgruppen - diamond interchange

	KanNr	Name	Beschreibung	Typ	TK	tMinF	tMinS	tFS	tSF
1	1	Kfz1	SB1	KFZ 3-feldig	1	5	2	3	
2	2	Kfz2	WB	KFZ 3-feldig	1	5	2	3	
3	3	Kfz3	NB1	KFZ 3-feldig	1	5	2	3	
4	4	Kfz4	EB	KFZ 3-feldig	1	5	2	3	
5	5	Kfz5	SBLT	KFZ 3-feldig	1	5	2	3	
6	6	Kfz6	SB	KFZ 3-feldig	1	5	2	3	
7	7	Kfz7	NBLT	KFZ 3-feldig	1	5	2	3	
8	8	Kfz8	NB	KFZ 3-feldig	1	5	2	3	
9	9	Kfz9	WBLT	KFZ 3-feldig	1	5	2	3	
10	10	Kfz10	EBLT	KFZ 3-feldig	1	5	2	3	

Screen 4.15. Define parameters for signal groups

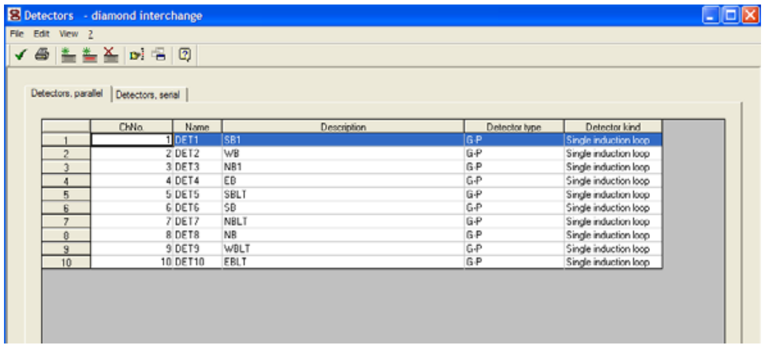
(b) Detectors

Parallel detectors are physical detectors connected to the controller by cables, while serial detectors are logical detectors extracted from a public transit (PT) detection system. This example uses parallel detectors.

The three detector types are:

- G-P: general purpose detectors,
- PT : public transit detectors and
- Ped: pedestrian detectors

Typically, the detector type is a single inductive loop.



Detectors - diamond interchange

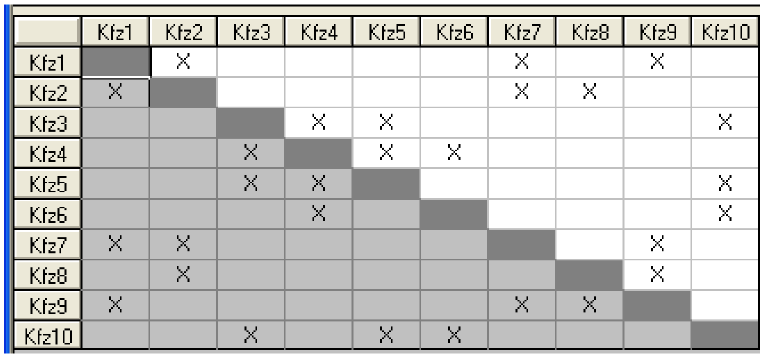
Detectors, parallel | Detectors, serial

	ChNo	Name	Description	Detector type	Detector kind
1	1	DET1	SB1	G-P	Single induction loop
2	2	DET2	WB	G-P	Single induction loop
3	3	DET3	NB1	G-P	Single induction loop
4	4	DET4	EB	G-P	Single induction loop
5	5	DET5	SBLT	G-P	Single induction loop
6	6	DET6	SB	G-P	Single induction loop
7	7	DET7	NBLT	G-P	Single induction loop
8	8	DET8	NB	G-P	Single induction loop
9	9	DET9	WBLT	G-P	Single induction loop
10	10	DET10	EBLT	G-P	Single induction loop

Screen 4.16. Define detectors

(c) Conflict Matrix

Define the “conflict matrix” of various signal groups.



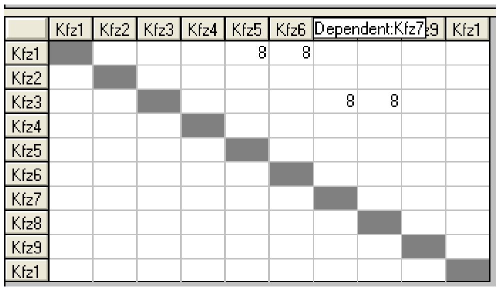
	Kfz1	Kfz2	Kfz3	Kfz4	Kfz5	Kfz6	Kfz7	Kfz8	Kfz9	Kfz10
Kfz1		X					X		X	
Kfz2	X						X	X		
Kfz3				X	X					X
Kfz4			X		X	X				
Kfz5			X	X						X
Kfz6				X						X
Kfz7	X	X							X	
Kfz8		X							X	
Kfz9	X						X	X		
Kfz10			X		X	X				

Screen 4.17. Define conflict matrix for signal groups

(d) Offset Matrix Start and Offset Matrix End

The “offset matrix start” and “offset matrix end” define the constant starting and ending offsets between different signal groups. It should be noted that these are not the offset between different intersections.

In this example, offset matrix start has been defined as follows:

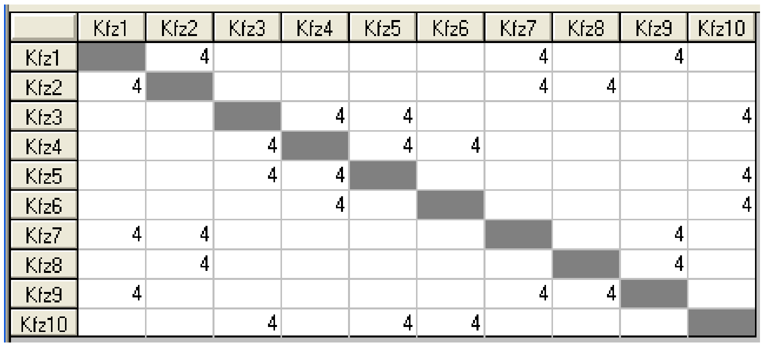


	Kfz1	Kfz2	Kfz3	Kfz4	Kfz5	Kfz6	Dependent:Kfz7		9	Kfz1
Kfz1					8	8				
Kfz2										
Kfz3							8	8		
Kfz4										
Kfz5										
Kfz6										
Kfz7										
Kfz8										
Kfz9										
Kfz1										

Screen 4.18. Define offset for signal groups

(e) Phase Change Matrix

The next step is to assign the clearance between signal groups.



	Kfz1	Kfz2	Kfz3	Kfz4	Kfz5	Kfz6	Kfz7	Kfz8	Kfz9	Kfz10
Kfz1	▇	4					4		4	
Kfz2	4	▇					4	4		
Kfz3			▇	4	4					4
Kfz4			4	▇	4	4				
Kfz5			4	4	▇					4
Kfz6				4		▇				4
Kfz7	4	4					▇		4	
Kfz8		4						▇	4	
Kfz9	4						4	4	▇	
Kfz10			4		4	4				▇

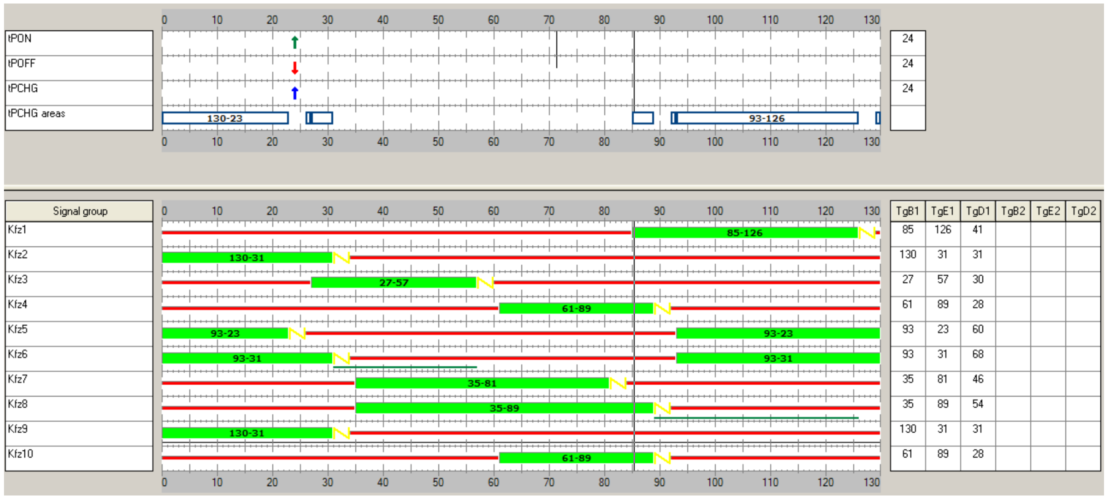
Screen 4.19. Define phase change matrix for signal groups

(f) Signal Timing Plan

The signal timing plan refers to the fixed time plan. One must first define a pointer for the program control.

- tPON: point of time for program switch on
- tPOFF: point of time for program switch off
- tPCHG: point of time for best program change

These pointers are defined for the real controller in order to determine when the program is switched off and on and when it is possible to switch between programs. In the Paramics plug-in, this function is not simulated.

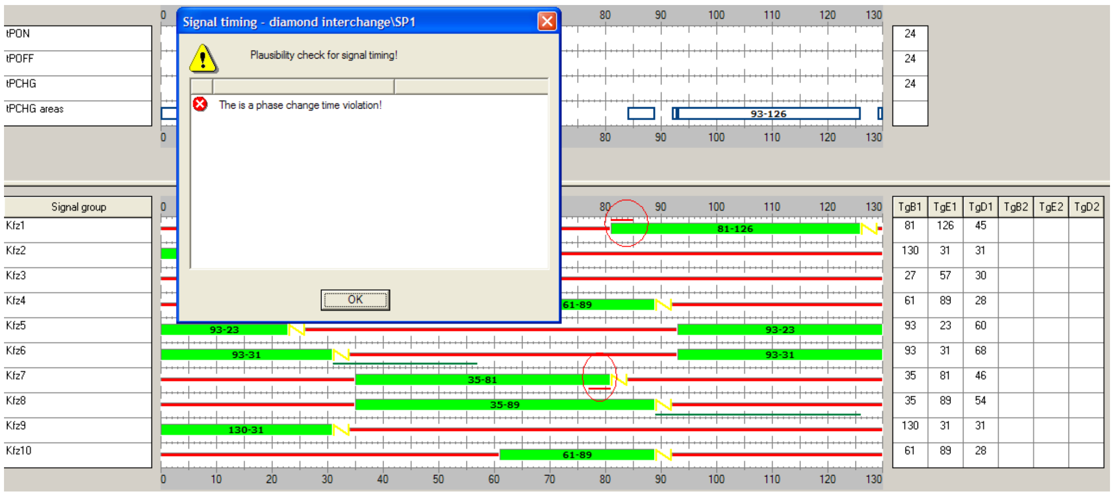


tPON	24
tPOFF	24
tPCHG	24
tPCHG areas	

Signal group	TgB1	TgE1	TgD1	TgB2	TgE2	TgD2
Kfz1	85	126	41			
Kfz2	130	31	31			
Kfz3	27	57	30			
Kfz4	61	89	28			
Kfz5	93	23	60			
Kfz6	93	31	68			
Kfz7	35	81	46			
Kfz8	35	89	54			
Kfz9	130	31	31			
Kfz10	61	89	28			

Screen 4.20. Define signal timing plan

When the signal timing plan is assigned, the program will check the signal plan using the phase change matrix. If they are not consistent with one another, the warning message shown below appears.


Signal timing - diamond interchange\SP1

Plausibility check for signal timing!

The is a phase change time violation!

OK

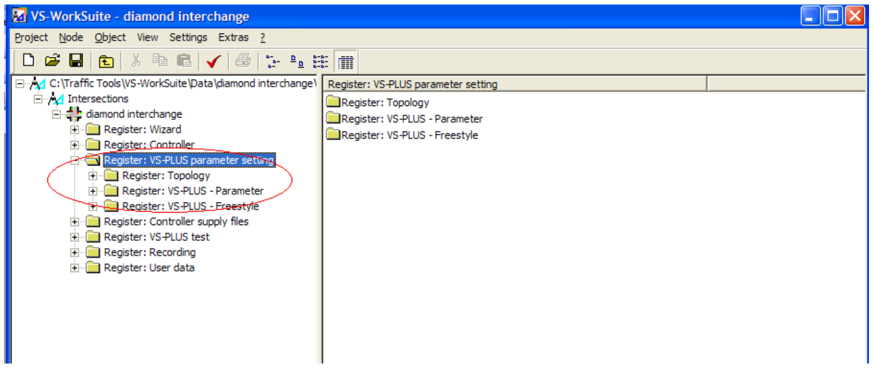
tPON	24
tPOFF	24
tPCHG	24
tPCHG areas	

Signal group	TgB1	TgE1	TgD1	TgB2	TgE2	TgD2
Kfz1	81	126	45			
Kfz2	130	31	31			
Kfz3	27	57	30			
Kfz4	61	89	28			
Kfz5	93	23	60			
Kfz6	93	31	68			
Kfz7	35	81	46			
Kfz8	35	89	54			
Kfz9	130	31	31			
Kfz10	61	89	28			

Screen 4.21. Error message in signal timing plan

(3) VS-PLUS Parameter Setting

Parameter setting is the core step in the VS-PLUS program. All the VS-PLUS parameters are defined in this step.


VS-WorkSuite - diamond interchange

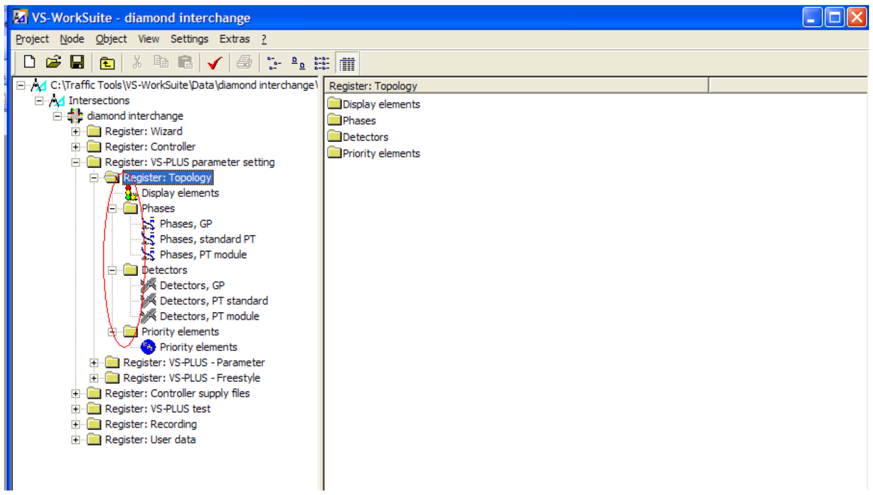
C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\
- Intersections
  - diamond interchange
    - Register: Wizard
    - Register: Controller
    - Register: VS-PLUS parameter setting
      - Register: Topology
      - Register: VS-PLUS - Parameter
      - Register: VS-PLUS - Freestyle
    - Register: Controller supply files
    - Register: VS-PLUS test
    - Register: Recording
    - Register: User data

Register: VS-PLUS parameter setting
- Register: Topology
- Register: VS-PLUS - Parameter
- Register: VS-PLUS - Freestyle

Screen 4.22. Parameters in "VS-PLUS parameter" groups

(a) VS-PLUS: Topology

The display element, phase, detectors, and priority element for VS-PLUS are defined.


VS-WorkSuite - diamond interchange

Project Node Object View Settings Extras ?

```
C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\
  Intersections
    diamond interchange
      Register: Wizard
      Register: Controller
      Register: VS-PLUS parameter setting
        Register: Topology
          Display elements
          Phases
            Phases, GP
            Phases, standard PT
            Phases, PT module
          Detectors
            Detectors, GP
            Detectors, PT standard
            Detectors, PT module
          Priority elements
            Priority elements
        Register: VS-PLUS - Parameter
        Register: VS-PLUS - Freestyle
      Register: Controller supply files
      Register: VS-PLUS test
      Register: Recording
      Register: User data
```

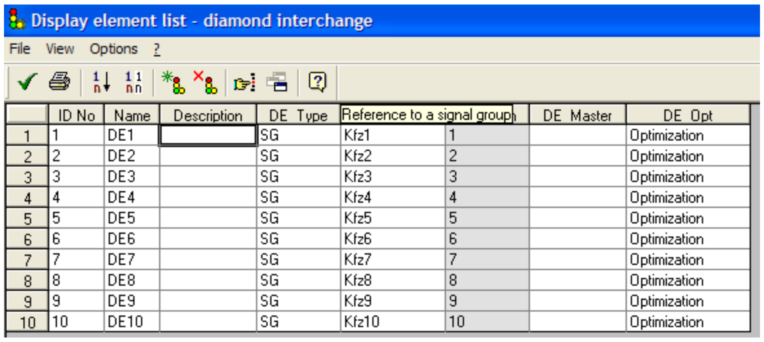
Register: Topology
- Display elements
- Phases
- Detectors
- Priority elements

Screen 4.23. Topology in "VS-PLUS parameter" groups

Please note that the VS-PLUS topology is not identical to the controller topology when the signal group, display element and phase are not on a 1:1 mapping scale.

(a.1) Display Elements

As shown in Screen 4.24, the table below defines the display elements and each display element is mapped with one signal group.


Display element list - diamond interchange

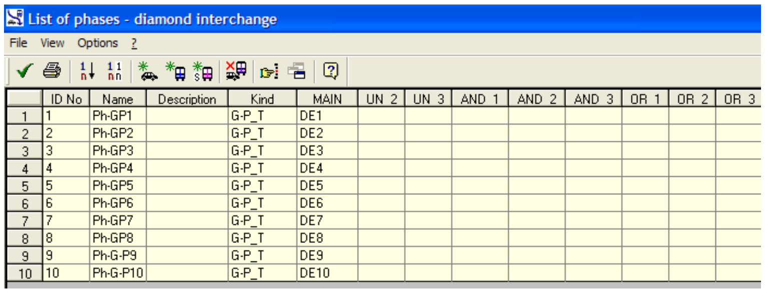
File View Options ?

	ID No	Name	Description	DE Type	Reference to a signal group		DE Master	DE Opt
1	1	DE1		SG	Kfz1	1		Optimization
2	2	DE2		SG	Kfz2	2		Optimization
3	3	DE3		SG	Kfz3	3		Optimization
4	4	DE4		SG	Kfz4	4		Optimization
5	5	DE5		SG	Kfz5	5		Optimization
6	6	DE6		SG	Kfz6	6		Optimization
7	7	DE7		SG	Kfz7	7		Optimization
8	8	DE8		SG	Kfz8	8		Optimization
9	9	DE9		SG	Kfz9	9		Optimization
10	10	DE10		SG	Kfz10	10		Optimization

Screen 4.24. Define the display elements

(a.2) Phases

In Screen 4.25, the table below is used to define phases. All phases are for general purpose traffic (G-P-T). Each phase corresponds to one display element which is defined in Screen 4.24.


List of phases - diamond interchange

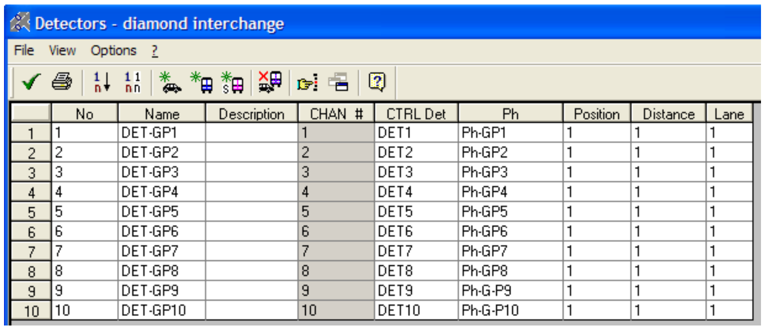
File View Options ?

	ID No	Name	Description	Kind	MAIN	UN 2	UN 3	AND 1	AND 2	AND 3	OR 1	OR 2	OR 3
1	1	Ph-GP1		G-P_T	DE1								
2	2	Ph-GP2		G-P_T	DE2								
3	3	Ph-GP3		G-P_T	DE3								
4	4	Ph-GP4		G-P_T	DE4								
5	5	Ph-GP5		G-P_T	DE5								
6	6	Ph-GP6		G-P_T	DE6								
7	7	Ph-GP7		G-P_T	DE7								
8	8	Ph-GP8		G-P_T	DE8								
9	9	Ph-G-P9		G-P_T	DE9								
10	10	Ph-G-P10		G-P_T	DE10								

Screen 4.25. Define the phases

(a.3) Detectors

In Screen 4.26, detectors are defined in this table. Please note that the parameters of "Position", "Distance", and "lane" are only informative. They are not being used at this stage.


Detectors - diamond interchange

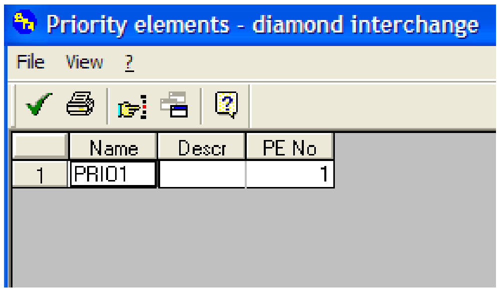
File View Options ?

	No	Name	Description	CHAN #	CTRL Det	Ph	Position	Distance	Lane
1	1	DET-GP1		1	DET1	Ph-GP1	1	1	1
2	2	DET-GP2		2	DET2	Ph-GP2	1	1	1
3	3	DET-GP3		3	DET3	Ph-GP3	1	1	1
4	4	DET-GP4		4	DET4	Ph-GP4	1	1	1
5	5	DET-GP5		5	DET5	Ph-GP5	1	1	1
6	6	DET-GP6		6	DET6	Ph-GP6	1	1	1
7	7	DET-GP7		7	DET7	Ph-GP7	1	1	1
8	8	DET-GP8		8	DET8	Ph-GP8	1	1	1
9	9	DET-GP9		9	DET9	Ph-G-P9	1	1	1
10	10	DET-GP10		10	DET10	Ph-G-P10	1	1	1

Screen 4.26. Define the detectors

(a.4) Priority elements

In this example, all of the phases are located in the same priority group; thus, only one priority element is defined.


Priority elements - diamond interchange

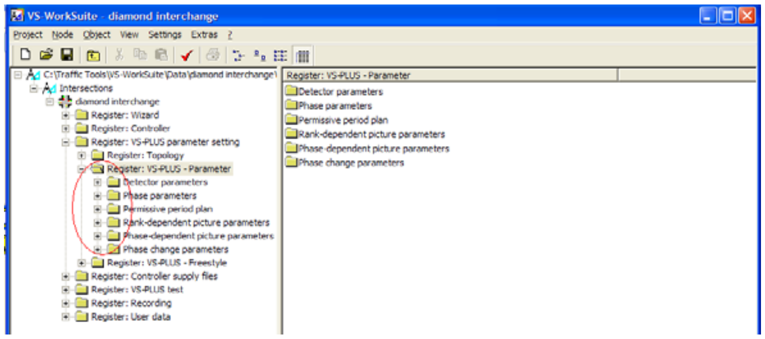
File View ?

	Name	Descr	PE No
1	PRIO1		1

Screen 4.27. Define the priority elements

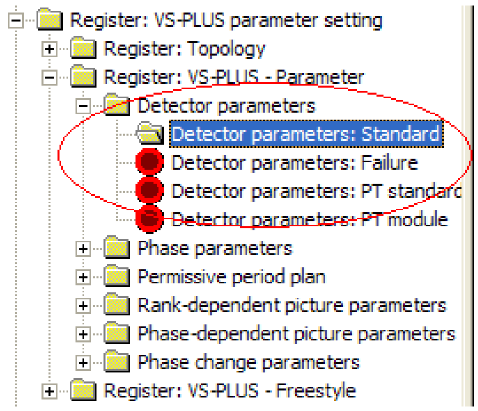
(b) VS-PLUS Parameters

Define the amount of detail needed for different detectors and phase parameters.


VS-WorkSuite - diamond interchange
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\
Intersections
diamond interchange
Register: Wizard
Register: Controller
Register: VS-PLUS parameter setting
Register: Topology
Register: VS-PLUS - Parameter
Detector parameters
Phase parameters
Permissive period plan
Rank-dependent picture parameters
Phase-dependent picture parameters
Phase change parameters
Register: VS-PLUS - Freestyle
Register: Controller supply files
Register: VS-PLUS test
Register: Recording
Register: User data
Register: VS-PLUS - Parameter
Detector parameters
Phase parameters
Permissive period plan
Rank-dependent picture parameters
Phase-dependent picture parameters
Phase change parameters

Screen 4.28. VS-PLUS parameters

(b.1) Detectors

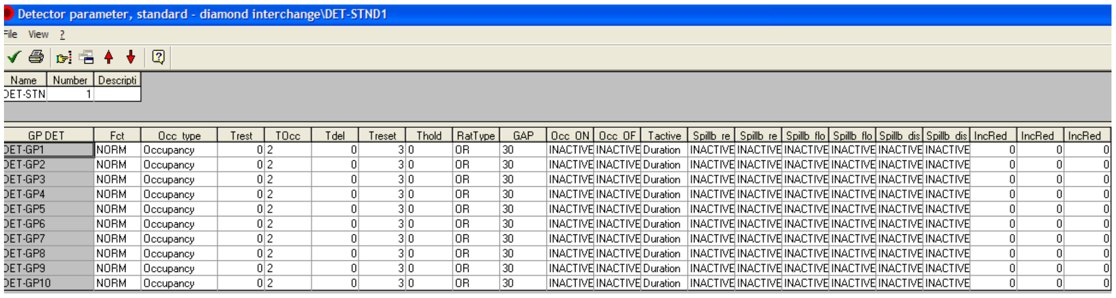

Register: VS-PLUS parameter setting
Register: Topology
Register: VS-PLUS - Parameter
Detector parameters
Detector parameters: Standard
Detector parameters: Failure
Detector parameters: PT standard
Detector parameters: PT module
Phase parameters
Permissive period plan
Rank-dependent picture parameters
Phase-dependent picture parameters
Phase change parameters
Register: VS-PLUS - Freestyle

Screen 4.29. Detector parameters

As shown in Screen 4.30, set the Occ_type column (detector call mode) to "Occupancy". The four detector call modes are:

- impulse,
- occupancy,
- impulse with occupancy and
- gap.

Users can select the type of detector call mode based on their real situation. The gap time can then be set to 30 (1/10 second). The gap time defines how long a detector continues calling the controller after having been left by a vehicle. Users can refer to the VS-PLUS user's manual Part 5 entitled "Detectors" for more detailed information regarding all detector parameters.

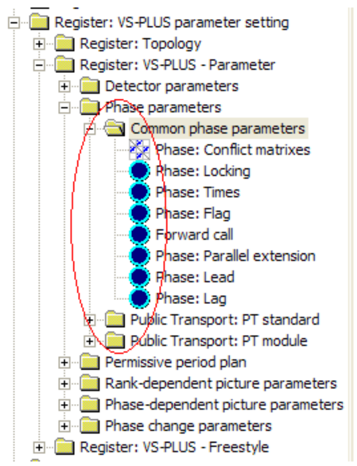

Detector parameter, standard - diamond interchange\DET-STND1
File View ?

Name	Number	Descripti
DET-STN	1	

GP DET	Fct	Occ type	Trest	TOcc	Tdel	Treset	Thold	RatType	GAP	Occ ON	Occ OF	Tactive	Spillb re	Spillb re	Spillb flo	Spillb flo	Spillb dis	Spillb dis	IncRed	IncRed	IncRed
DET-GP1	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP2	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP3	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP4	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP5	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP6	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP7	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP8	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP9	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0
DET-GP10	NORM	Occupancy	0	2	0	3	0	OR	30	INACTIVE	INACTIVE	Duration	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	0	0	0

Screen 4.30. Standard detector parameters

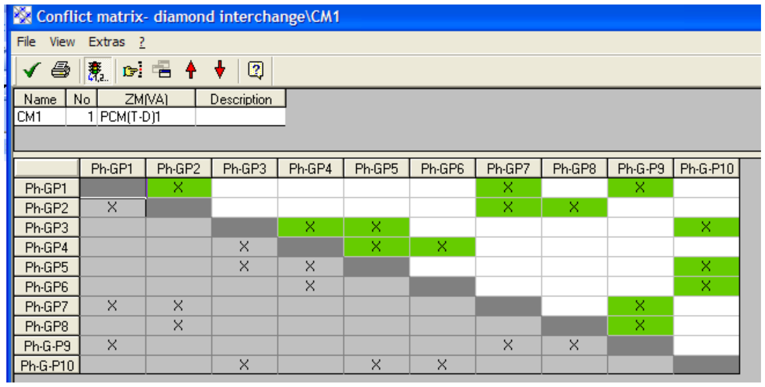
(b.2) Phase Parameters


Register: VS-PLUS parameter setting
Register: Topology
Register: VS-PLUS - Parameter
Detector parameters
Phase parameters
Common phase parameters
Phase: Conflict matrixes
Phase: Locking
Phase: Times
Phase: Flag
Forward call
Phase: Parallel extension
Phase: Lead
Phase: Lag
Public Transport: PT standard
Public Transport: PT module
Permissive period plan
Rank-dependent picture parameters
Phase-dependent picture parameters
Phase change parameters
Register: VS-PLUS - Freestyle

Screen 4.31. Phase parameters

<u>Conflict Matrix</u>

Since phases and signal groups are mapped on a 1:1 scale in this example, the conflict matrix is identical to the signal groups' conflict matrix as defined in the Controller section of this report.



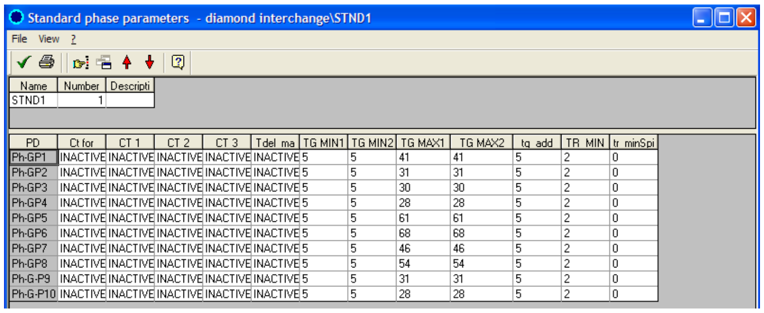
Name	No	ZM(VA)	Description
CM1	1	PCM(T-D)1	

	Ph-GP1	Ph-GP2	Ph-GP3	Ph-GP4	Ph-GP5	Ph-GP6	Ph-GP7	Ph-GP8	Ph-G-P9	Ph-G-P10
Ph-GP1		X					X		X	
Ph-GP2	X						X	X		
Ph-GP3				X	X					X
Ph-GP4			X		X	X				
Ph-GP5			X	X						X
Ph-GP6				X						X
Ph-GP7	X	X							X	
Ph-GP8		X							X	
Ph-G-P9	X						X	X		
Ph-G-P10			X		X	X				

Screen 4.32. Phase change matrix

<u>Phase Times</u>

As shown in Screen 4.33, according to the time sheet, one must input the corresponding values to each phase time parameter.



Name	Number	Descripti
STND1	1	

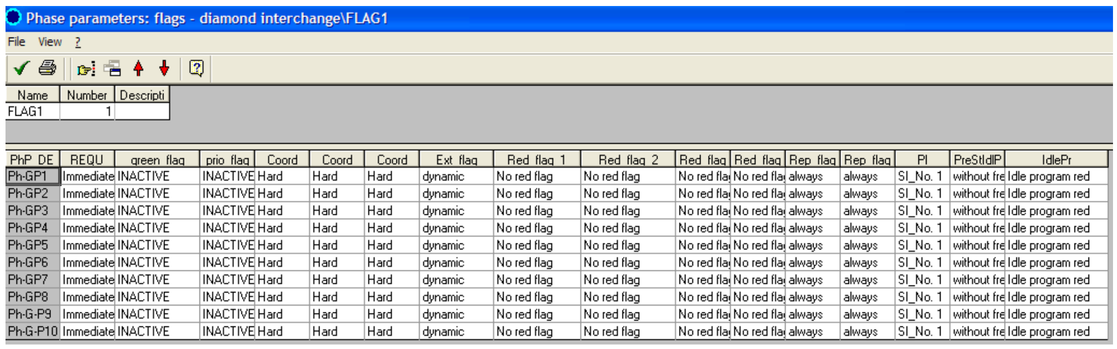
PD	Ct for	CT 1	CT 2	CT 3	Tdel ma	TG MIN1	TG MIN2	TG MAX1	TG MAX2	tg add	TR MIN	tr minSpi
Ph-GP1	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	41	41	5	2	0
Ph-GP2	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	31	31	5	2	0
Ph-GP3	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	30	30	5	2	0
Ph-GP4	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	28	28	5	2	0
Ph-GP5	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	61	61	5	2	0
Ph-GP6	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	68	68	5	2	0
Ph-GP7	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	46	46	5	2	0
Ph-GP8	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	54	54	5	2	0
Ph-G-P9	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	31	31	5	2	0
Ph-G-P10	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	28	28	5	2	0

Screen 4.33. Phase Times

Phase Flag

By setting the phase flag (i.e., green flag, red flag, or coordination flag), the users can control individual phases of the interchange. If needed, detailed information about the setting of these detector parameters can be found in the VS-PLUS user's manual, Part 6 entitled "Phase."

Please note that the coordination flag mentioned in Screen 4.34 is different with the general concept of coordination. It defines how strongly the permissive periods have to be respected by the green commands, i.e., whether the permissive periods have to be followed exactly or if there is a certain degree of flexibility.



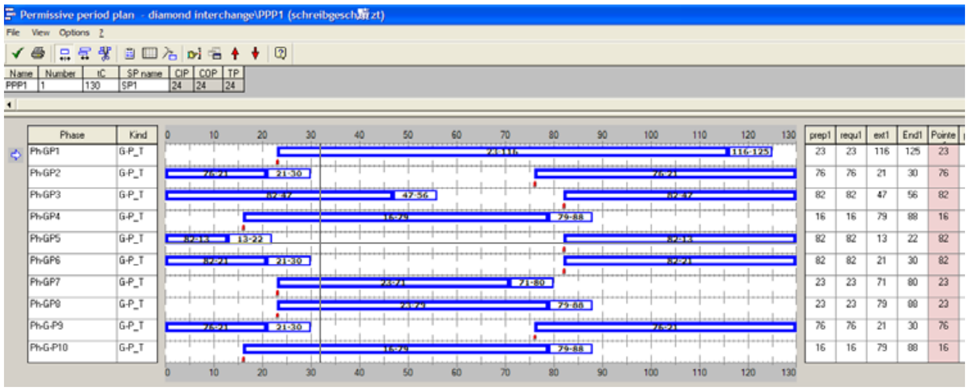
Name	Number	Descripti
FLAG1	1	

PhP DE	REQU	green flag	prio flag	Coord	Coord	Coord	Ext flag	Red flag 1	Red flag 2	Red flag	Red flag	Rep flag	Rep flag	PI	PreStIdlP	IdlePr
Ph-GP1	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP2	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP3	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP4	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP5	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP6	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP7	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-GP8	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-G-P9	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red
Ph-G-P10	Immediate	INACTIVE	INACTIVE	Hard	Hard	Hard	dynamic	No red flag	No red flag	No red flaı	No red flaı	always	always	SI_No. 1	without fre	Idle program red

Screen 4.34. Phase flag

(b.3) Permissive Period Plan

The permissive period plan is the most critical concept when using the VS-PLUS simulation software. In Screen 4.35, the initial permissive period plan is generated by using the signal timing plan.



Permissive period plan - diamond interchange\PPP1 (schreibgesch zt)

Name	Number	tC	SP name	CIP	COP	TP
PPP1	1	130	SP1	24	24	24

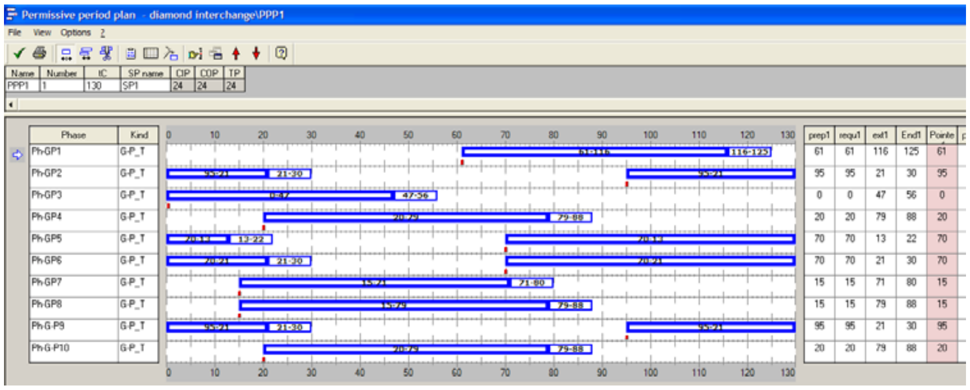
Phase	Kind	prep1	requ1	ext1	End1	Pointe
Ph-GP1	G-P_T	23	23	116	125	23
Ph-GP2	G-P_T	76	76	21	30	76
Ph-GP3	G-P_T	82	82	47	56	82
Ph-GP4	G-P_T	16	16	79	88	16
Ph-GP5	G-P_T	82	82	13	22	82
Ph-GP6	G-P_T	82	82	21	30	82
Ph-GP7	G-P_T	23	23	71	80	23
Ph-GP8	G-P_T	23	23	79	88	23
Ph-G-P9	G-P_T	76	76	21	30	76
Ph-G-P10	G-P_T	16	16	79	88	16

Screen 4.35. Initial permissive period plan

Each phase is then divided into two ranges—the call range and the extension range. The permissive signal is the frame for all VS-PLUS operations. The following is the brief interpretation of the titles used in the program:

• "requ"	is the calling start point.
• "prep"	is used for PT priority issues in order to give the advanced call range. In this example, "prep" and "requ" should always have the same value.
• "ext"	is the extension start point.
• "end"	is the frame end point.
• "pointer"	is used to define the sequence of the phases.

According to this example, the permissive period plan can be modified as follows.



Permissive period plan - diamond interchange\PPP1

Name	Number	tC	SP name	CIP	COP	TP
PPP1	1	130	SP1	24	24	24

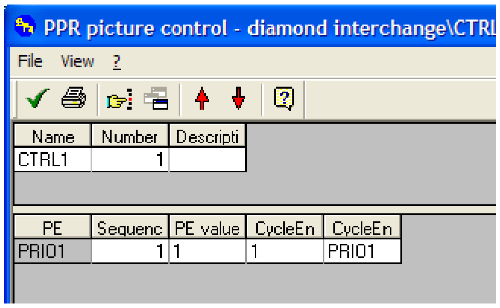
Phase	Kind	prep1	requ1	ext1	End1	Pointe
Ph-GP1	G-P_T	61	61	116	125	61
Ph-GP2	G-P_T	95	95	21	30	95
Ph-GP3	G-P_T	0	0	47	56	0
Ph-GP4	G-P_T	20	20	79	88	20
Ph-GP5	G-P_T	70	70	13	22	70
Ph-GP6	G-P_T	70	70	21	30	70
Ph-GP7	G-P_T	15	15	71	80	15
Ph-GP8	G-P_T	15	15	79	88	15
Ph-G-P9	G-P_T	95	95	21	30	95
Ph-G-P10	G-P_T	20	20	79	88	20

Screen 4.36. Permissive period plan

(b.4) Rank-dependent Picture Parameters

Picture Control

As stated previously in this example, there is only one priority element.



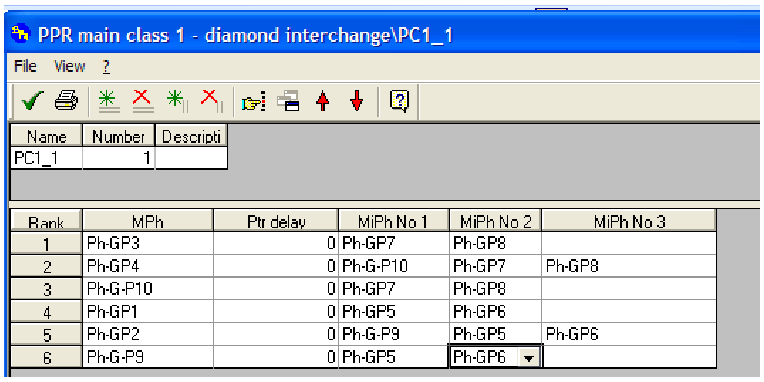
Name	Number	Descripti
CTRL1	1	

PE	Sequenc	PE value	CycleEn	CycleEn
PRIO1	1	1	1	PRIO1

Screen 4.37. Picture control

Main Series Class

Next, define the main series with the minor series.



Name	Number	Descripti
PC1_1	1	

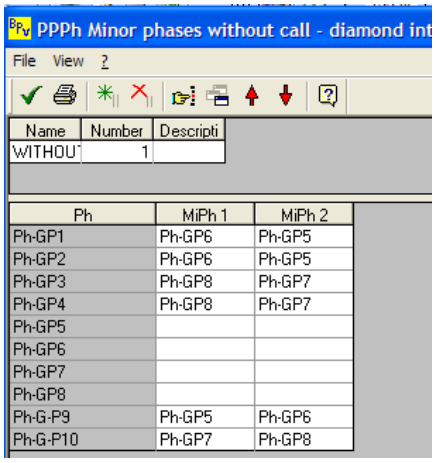
Rank	MPh	Ptr delay	MiPh No 1	MiPh No 2	MiPh No 3
1	Ph-GP3	0	Ph-GP7	Ph-GP8	
2	Ph-GP4	0	Ph-G-P10	Ph-GP7	Ph-GP8
3	Ph-G-P10	0	Ph-GP7	Ph-GP8	
4	Ph-GP1	0	Ph-GP5	Ph-GP6	
5	Ph-GP2	0	Ph-G-P9	Ph-GP5	Ph-GP6
6	Ph-G-P9	0	Ph-GP5	Ph-GP6	

Screen 4.38. Main series

(b.5) Phase-dependent Picture Parameters

Minor Phases without Call

Define the phases which are called automatically with another phase. For example, when phase1 receives green, phase 5 and phase 6 are called automatically.


PPPh Minor phases without call - diamond int

Name	Number	Descripti
WITHOU1	1	

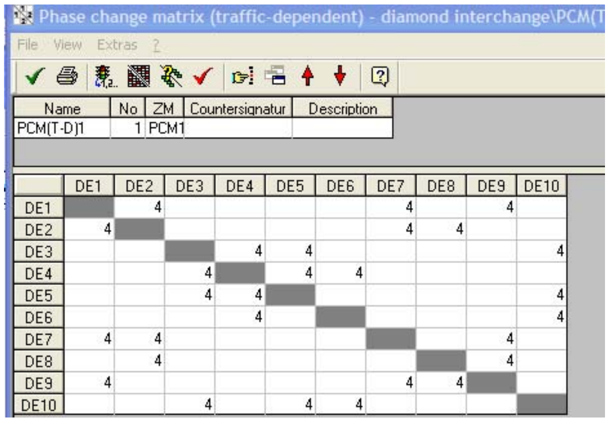
Ph	MiPh 1	MiPh 2
Ph-GP1	Ph-GP6	Ph-GP5
Ph-GP2	Ph-GP6	Ph-GP5
Ph-GP3	Ph-GP8	Ph-GP7
Ph-GP4	Ph-GP8	Ph-GP7
Ph-GP5		
Ph-GP6		
Ph-GP7		
Ph-GP8		
Ph-G-P9	Ph-GP5	Ph-GP6
Ph-G-P10	Ph-GP7	Ph-GP8

Screen 4.39. Minor phases without call

(b.6) Phase Change Parameters

In this example, the VS-PLUS phase change parameters are identical to the controller phase change parameters since there is only one phase per signal group.

<u>Phase Change Matrix</u>


Phase change matrix (traffic-dependent) - diamond interchange\PCM(T

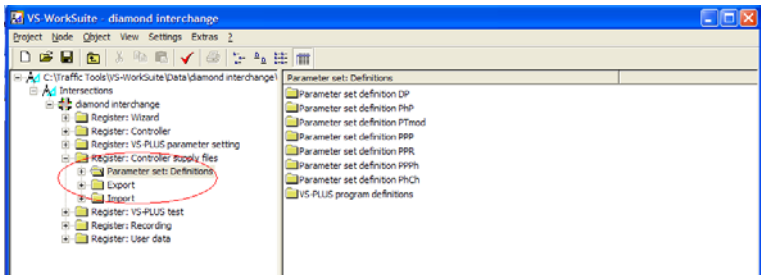
Name	No	ZM	Countersignatur	Description
PCM(T-D)1	1	PCM1		

	DE1	DE2	DE3	DE4	DE5	DE6	DE7	DE8	DE9	DE10
DE1	▨	4					4		4	
DE2	4	▨					4	4		
DE3			▨	4	4					4
DE4			4	▨	4	4				
DE5			4	4	▨					4
DE6				4		▨				4
DE7	4	4					▨		4	
DE8		4						▨	4	
DE9	4						4	4	▨	
DE10			4		4	4				▨

Screen 4.40. Phase change parameters

(4) Parameter Set Definitions

Because users are allowed to define multiple signal control plans (i.e., the user can define multiple sets of parameters such as permissive period plans, phase flags, phase times, etc); therefore, the users must define which set of parameters will be used in the VS-PLUS simulation. In this example, there is only one signal control plan.

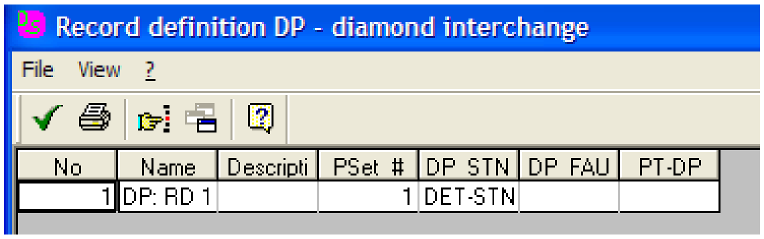

VS-WorkSuite - diamond interchange
Project Node Object View Settings Extras ?

C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\
- Intersections
  - diamond interchange
    - Register: Wizard
    - Register: Controller
    - Register: VS-PLUS parameter setting
    - Register: Controller supply files
      - Parameter set: Definitions
      - Export
      - Import
    - Register: VS-PLUS test
    - Register: Recording
    - Register: User data

Parameter set: Definitions
- Parameter set definition DP
- Parameter set definition PhP
- Parameter set definition PTmod
- Parameter set definition PPP
- Parameter set definition PPR
- Parameter set definition PPPh
- Parameter set definition PhCh
- VS-PLUS program definitions

Screen 4.41. Parameter Set Definitions

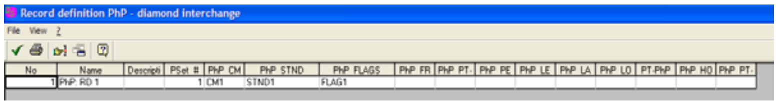
(a) Detector Parameters



No	Name	Descripti	PSet #	DP STN	DP FAU	PT-DP
1	DP: RD 1		1	DET-STN		

Screen 4.42. Define parameter set for detector

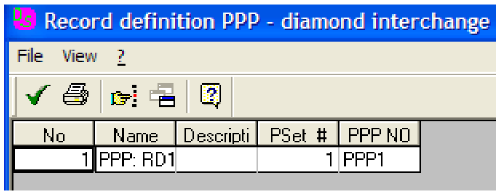
(b) Phase Parameters (PhP)



No	Name	Descripti	PSet #	PhP CM	PhP STND	PhP FLAGS	PhP FR	PhP PT-	PhP PE	PhP LE	PhP LA	PhP LO	PT-PhP	PhP HO	PhP PT-
1	PhP: RD 1		1	CM1	STND1	FLAG1									

Screen 4.43. Define parameter set for phase

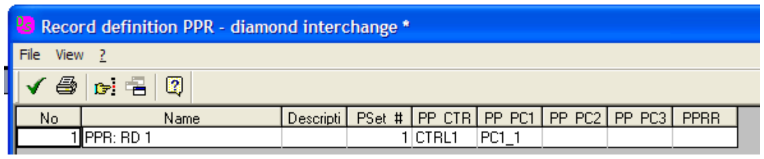
(c) Permissive Period Plan (PPP)



No	Name	Descripti	PSet #	PPP NO
1	PPP: RD1		1	PPP1

Screen 4.44. Define parameter set for permissive period plan

(d) Picture Parameters, Rank-dependent (PPR)

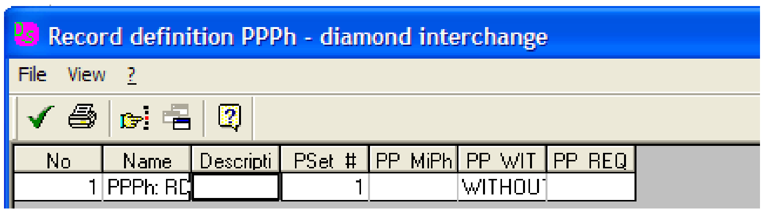


Record definition PPR - diamond interchange *

No	Name	Descripti	PSet #	PP CTR	PP PC1	PP PC2	PP PC3	PPRR
1	PPR: RD 1		1	CTRL1	PC1_1			

Screen 4.45. Define parameter set for permissive period plan

(e) Picture Parameters, Rank-dependent (PPPh)

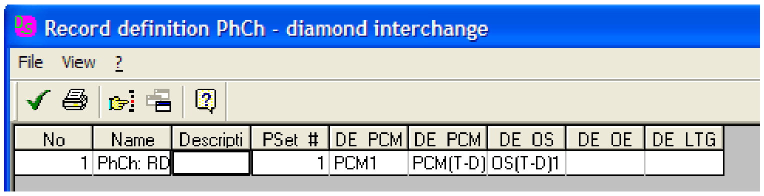


Record definition PPPh - diamond interchange

No	Name	Descripti	PSet #	PP MiPh	PP WIT	PP REQ
1	PPPh: RD		1		WITHOU	

Screen 4.46. Define parameter set for rank-dependent phase

(f) Phase Change Parameters (PhCh)

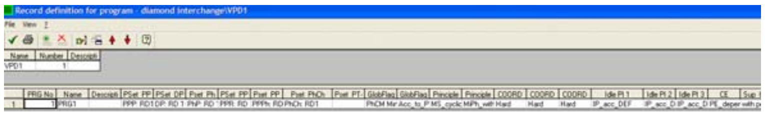


Record definition PhCh - diamond interchange

No	Name	Descripti	PSet #	DE PCM	DE PCM	DE OS	DE OE	DE LTG
1	PhCh: RD		1	PCM1	PCM(T-D)	OS(T-D)1		

Screen 4.47. Define parameter set for phase change parameters

(g) VS-PLUS Program Definitions



Record definition for program - diamond interchange\VPD1

Name	Number	Descripti
VPD1	1	

	PRG No	Name	Descripti	PSet PP	PSet DP	Pset Ph	PSet PP	Pset PP	Pset PhCh	Pset PT-	GlobFlag	GlobFlag	Principle	Principle	COORD	COORD	COORD	Idle PI 1	Idle PI 2	Idle PI 3	CE	Sup
1	1	PRG1		PPP: RD1	DP: RD 1	PhP: RD	PPR: RD	PPPh: RD	PhCh: RD1		PhCM Mir	Acc_to_P	MS_cyclic	MiPh_with	Hard	Hard	Hard	IP_acc_DEF	IP_acc_D	IP_acc_D	PE_deper	with p

Screen 4.48. Define parameter set for VS-PLUS program definitions

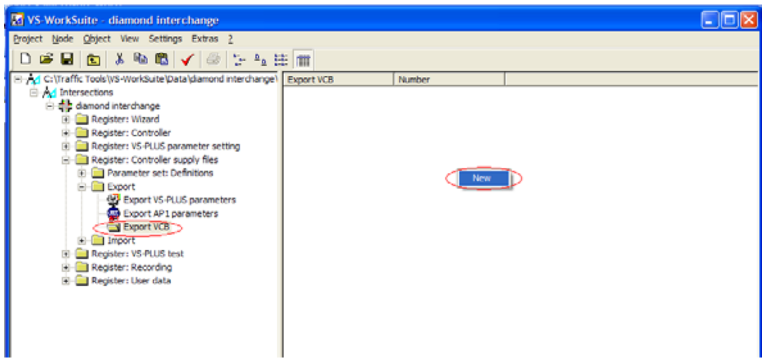
(5) Parameters Files Generation

After the parameters setting processed is finished, the VS-PLUS parameters can be exported to controller or to the parameters files which will be used by a microscopic simulation.

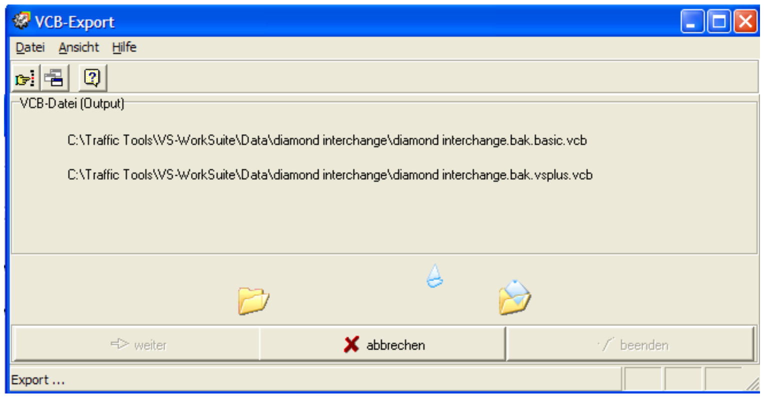
Currently, there are two types of parameters files:

- VCE: parameters files for VISSIM
- VCB: parameters files for controller or VS-Emulator

The Paramics/VS-PLUS simulationl uses the “VCB” parameters file.


VS-WorkSuite - diamond interchange
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\
Intersections
diamond interchange
Register: Wizard
Register: Controller
Register: VS-PLUS parameter setting
Register: Controller supply files
Parameter set: Definitions
Export
Export VS-PLUS parameters
Export AP1 parameters
Export VCB
Import
Register: VS-PLUS test
Register: Recording
Register: User data
Export VCB
Number
New

Screen 4.49. Export VS-PLUS parameters



C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\diamond interchange.bak.basic.vcb

C:\Traffic Tools\VS-WorkSuite\Data\diamond interchange\diamond interchange.bak.vsplus.vcb

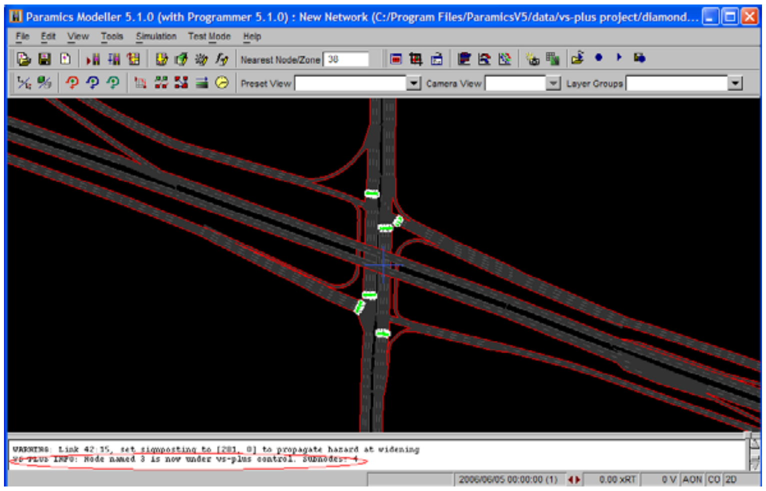
Screen 4.50. Export VS-PLUS parameters to the parameter files

Two parameter files—the “diamond interchange.basic.vcb” and the “diamond interchange.vsplus.vcb”—must be created in the project directory. These two parameter files will be used to convert the signal control parameters of VS-PLUS to Paramics. The parameters developed in VS-PLUS are now ready to be used in Paramics.

Paramics Simulation Model

The first step in creating a Paramics Simulation Model is to code the Paramics network. In order to do this, a road network must be constructed by adding nodes, links, zones, as well as coding detailed lanes and junction descriptions. The geometry of the network was derived from the aerial photo in this example, which is shown in Screen 4.51.

In addition, demand matrices from previously obtained origin/destination data must be constructed, and the movement and lane allocation must be defined. In this study, we used real traffic data from field observations for calibrating the simulation model. In this section, we introduce how the numerous signal timing parameters developed in VS-PLUS can be used in Paramics.


Paramics Modeller 5.1.0 (with Programmer 5.1.0) : New Network (C:/Program Files/ParamicsV5/data/vs-plus project/diamond...
File Edit View Tools Simulation Test Mode Help
Nearest Node/Zone 38
Preset View
Camera View
Layer Groups

```
WARNING: Link 42:15, set signposting to [281, 0] to propagate hazard at widening
VS-PLUS INFO: Node named 3 is now under vs-plus control. Subnodes: 4
```

2006/06/05 00:00:00 (1)
0.00 xRT
0 V
AON
CO
2D

Screen 4.51. Paramics network

Paramics/VS-PLUS Interface

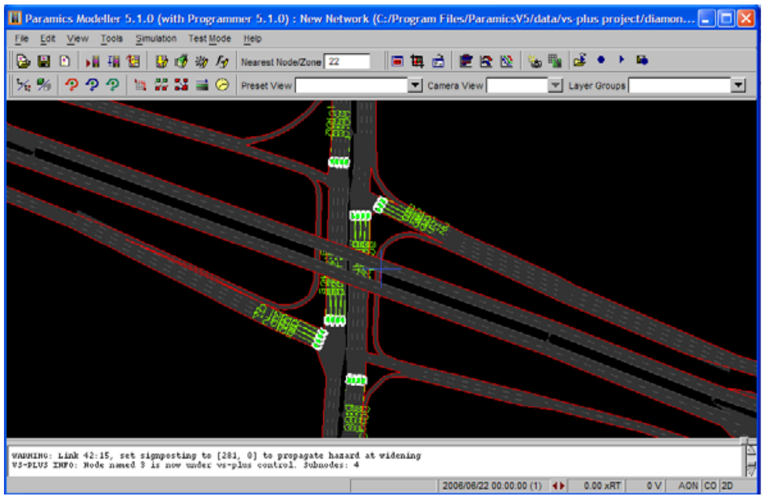
(1) Detectors Setting

Rules for labeling the detector's are as follows.

(a) The first number represents the intersection node number (node number in VS-PLUS).

(b) The last number represents the detector channel ID number.

For example: The label 1.6 corresponds to the detector that belongs to intersection node 1, and its channel ID number is 6. If more than one detector is to be used according to the same movement, they must be mapped into the same detector input channel in a manner such as 1.1.6 and 1.2.6. These multiple detectors will be treated as one detector in VS-PLUS.


Paramics Modeller 5.1.0 (with Programmer 5.1.0) : New Network (C:/Program Files/ParamicsV5/data/vs-plus project/diamon...
File Edit View Tools Simulation Test Mode Help
Nearest Node/Zone 22
Preset View
Camera View
Layer Groups

```
WARNING: Link 42:15, set signposting to [281, 0] to propagate hazard at widening
VS-PLUS INFO: Node named 3 is now under vs-plus control. Subnodes: 4
```

2006/06/22 00:00:00 (1)
0.00 xRT
0 V
AON
CO
2D

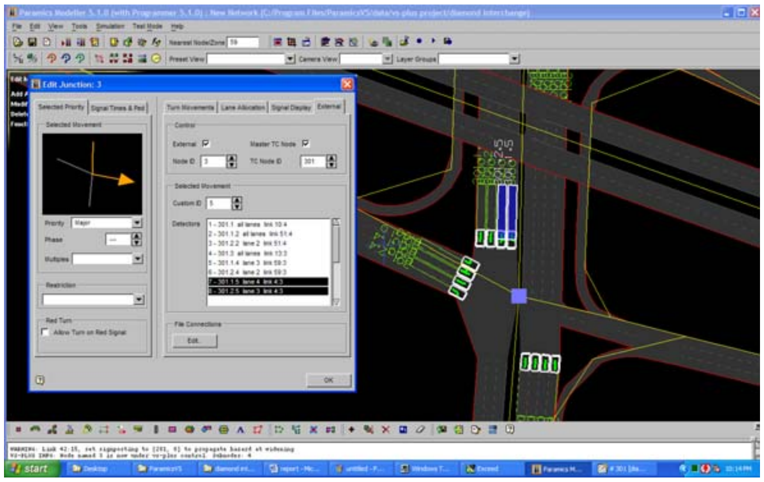
Screen 4.52. Define detectors in Paramics

(2) External Signal Setting

Rules governing the external signal definitions are:

(a) The traffic controller (TC) node ID must match the intersection node ID in VS-PLUS. In this example, the VS-PLUS node number is 301. The master controller node is located at the south junction.

(b) When selecting movement paths, the customer movement ID must match the detector channel ID number. For example, customer movement 2 must be associated with the detector whose channel ID is 2.


Edit Junction: 3

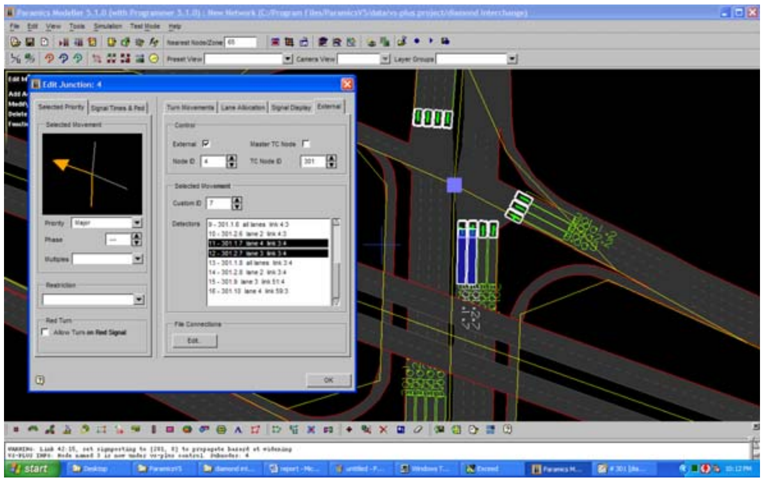
External ☑ Master TC Node ☑
Node ID 3 TC Node ID 301

Selected Movement
Custom ID 5
Priority Major

Detectors
1 - 301.1 all lanes link 10:4
2 - 301.1.2 all lanes link 51:4
3 - 301.2.2 lane 2 link 51:4
4 - 301.3 all lanes link 13:3
5 - 301.1.4 lane 3 link 59:3
6 - 301.2.4 lane 2 link 59:3
7 - 301.1.5 lane 4 link 4:3
8 - 301.2.5 lane 3 link 4:3

```
WARNING: Link 42:15, set signposting to [201, 0] to propagate hazard at widening
VS-PLUS INFO: Node named 3 is now under vs-plus control. #Nodes: 4
```

Screen 4.53. Define the movement-1


Edit Junction: 4

External ☑ Master TC Node ☐
Node ID 4 TC Node ID 301

Selected Movement
Custom ID 7
Priority Major

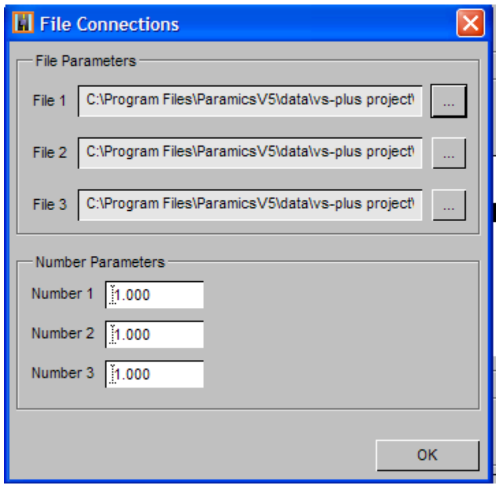
Detectors
9 - 301.1.6 all lanes link 4:3
10 - 301.2.6 lane 2 link 4:3
11 - 301.1.7 lane 4 link 3:4
12 - 301.2.7 lane 3 link 3:4
13 - 301.1.8 all lanes link 3:4
14 - 301.2.8 lane 2 link 3:4
15 - 301.9 lane 3 link 51:4
16 - 301.10 lane 4 link 59:3

```
WARNING: Link 42:15, set signposting to [201, 0] to propagate hazard at widening
VS-PLUS INFO: Node named 3 is now under vs-plus control. #Nodes: 4
```

Screen 4.54. Define the movement-2

(c) Define the File Connections:

First, click the Edit button in File Connections dialog box.
In the File Parameters dialog box:
Choose "VSPLUS_Emulator.exe" for File 1;
Choose "example1.basic.vcb" for File 2;
Choose "example1.vsplus.vcb" for File 3.
In the Number Parameters dialog box, input 1.0 for all.


File Connections

File Parameters

File 1	C:\Program Files\ParamicsV5\data\vs-plus project\
File 2	C:\Program Files\ParamicsV5\data\vs-plus project\
File 3	C:\Program Files\ParamicsV5\data\vs-plus project\

Number Parameters

Number 1	1.000
Number 2	1.000
Number 3	1.000

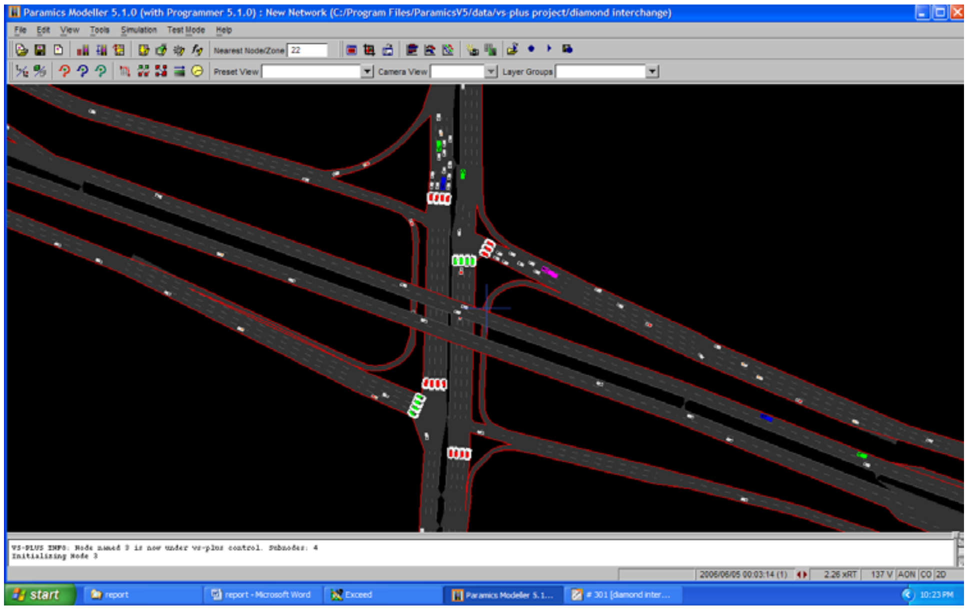
Screen 4.55. Build the files connection

(3) Plug-ins Files Setting

Define the directory of VS-PLUS plug-ins file “VSPLUSServerModeller.dll” in file “programming.modeller”.

Evaluation

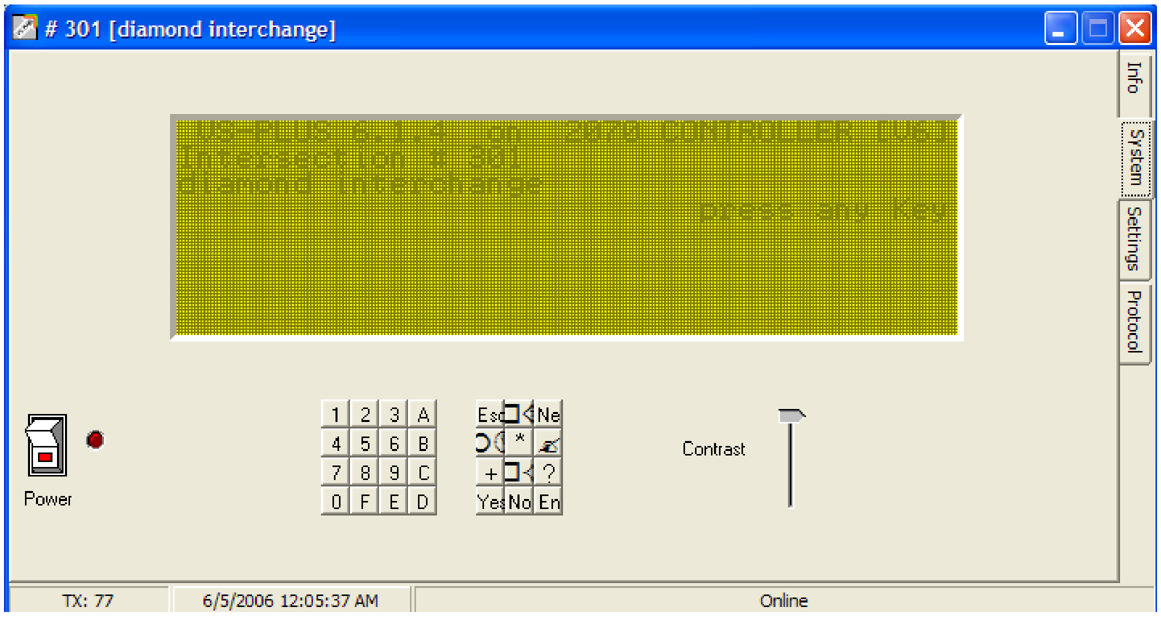
In the simulation, the signals are now controlled by VS-PLUS. Figure 3.56 is a snapshot of the Paramics simulation running with a VS-PLUS controller at the test diamond interchange.


Paramics Modeller 5.1.0 (with Programmer 5.1.0) : New Network (C:/Program Files/ParamicsV5/data/vs-plus project/diamond interchange)

```
VS-PLUS INFO: Node named 3 is now under vs-plus control. Subnodes: 4
Initializing Node 3
```

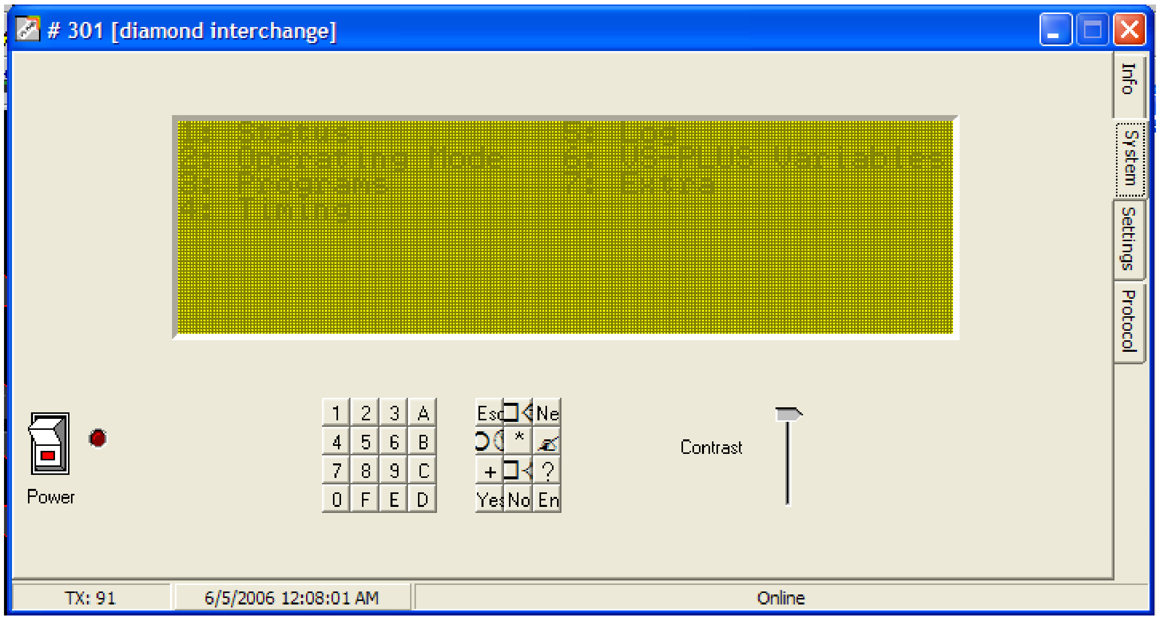
Screen 4.56. Running in simulation

VS-Emulator is the VS-PLUS on-line monitor or virtual controller. It can visualize the main control parameters on the screen. The control parameters can be modified during the simulation runs.



```
VS-PLUS 6.1.4   on   2070 CONTROLLER [US]
Intersection # 301
diamond interchange
                           press any key
```

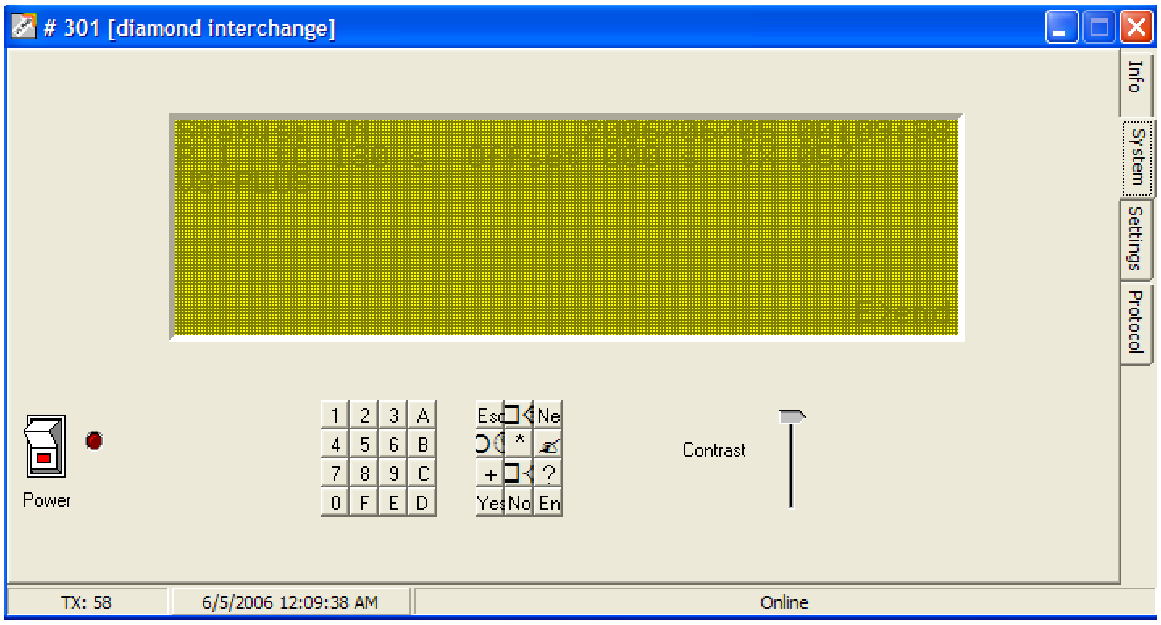
Screen 4.57. VS-Emulator



```
1:  Status              5:  Log
2:  Operating Mode      6:  VS-PLUS Variables
3:  Programs            7:  Extra
4:  Timing
```

Screen 4.58. VS-Emulator control panel

The following figure shows the current status of the controller, the current controller local time and the system time.

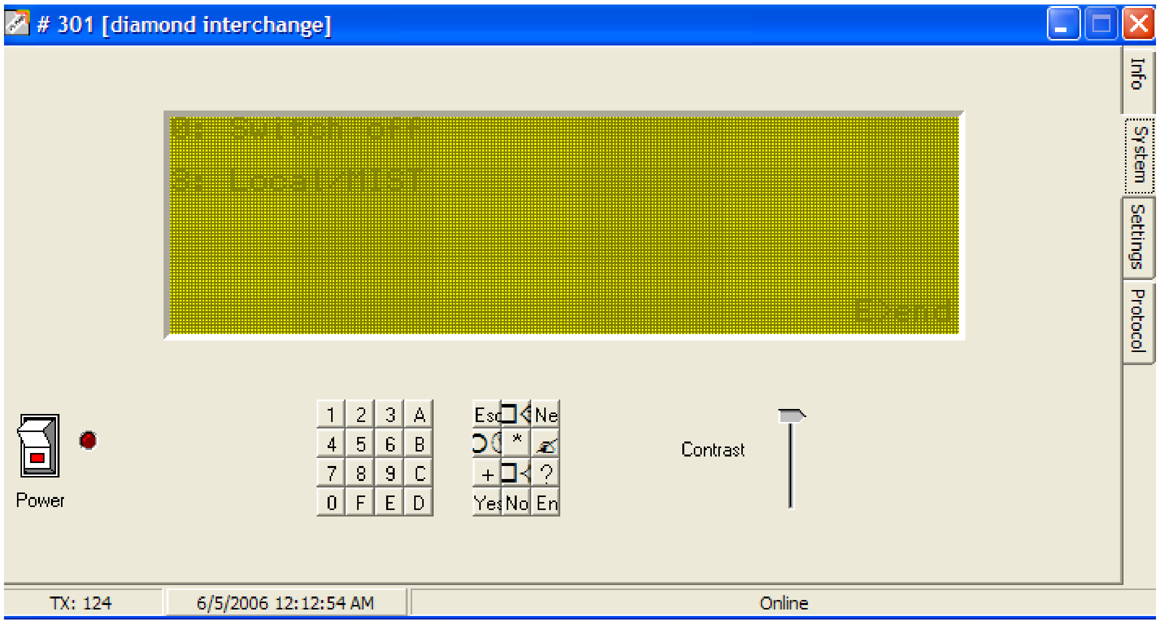


```
Status: ON          2006/06/05 00:09:38
P 1  TC 130 s  Offset 000 s  TX 057
US-PLUS
                                   E:end
```

TX: 58 | 6/5/2006 12:09:38 AM | Online

Screen 4.59. Status of the controller

The users can switch the controller on/off by operating the virtual controller.



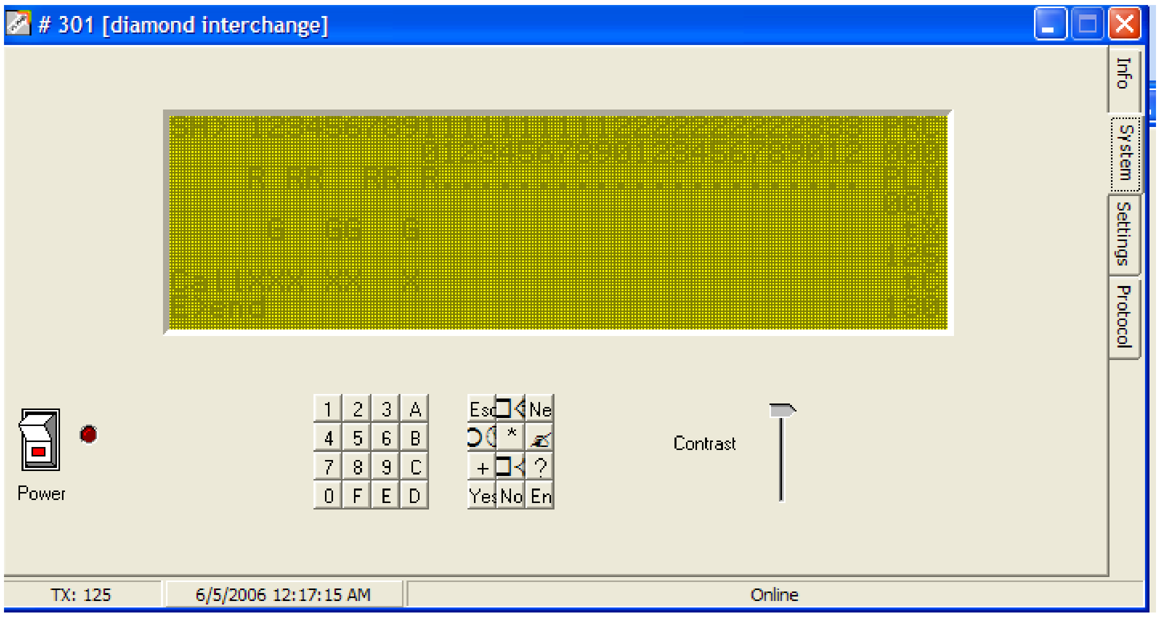
```
0: Switch off

8: Local/MIST
                                   E:end
```

TX: 124 | 6/5/2006 12:12:54 AM | Online

Screen 4.60. Switch on/off

Users can also check the phase status. The following figure shows all of the status of all signal phases (green, yellow, red and call).



Screen 4.61. Status of the phases

4.2.3 Isolated Fully Actuated Intersection

A beginner may have found it difficult to correctly set all of the needed parameters without understanding the principles set forth in the program. Fortunately, VS-PLUS provides a helpful tool in the form of the VS-NEMA wizard (Verkehrs-Systeme 2004). In this section, the example below illustrates how to design an intersection signal control based on the National Electrical Manufacturers Association (NEMA) standard.

The VS-NEMA Wizard is a user friendly tool that designs a signal controller based on the NEMA standards. As shown in Figure 4.9, the operation of VS-NEMA Wizard consists of seven steps.

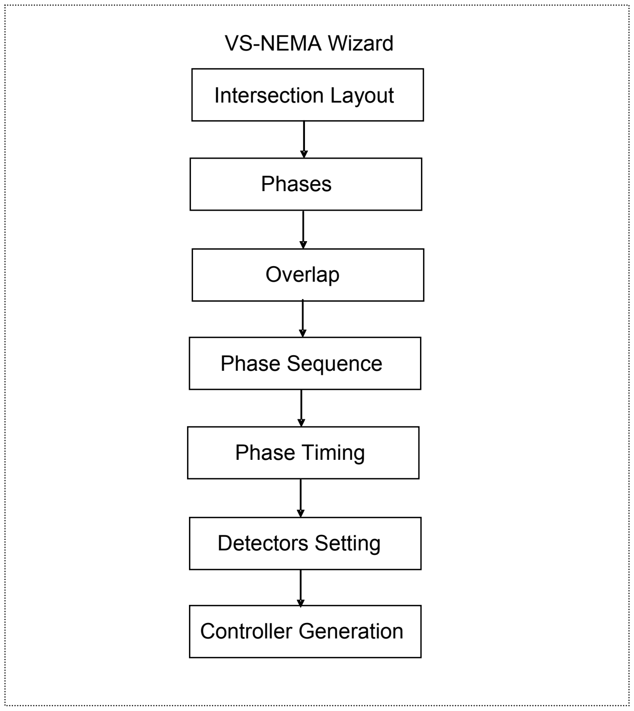
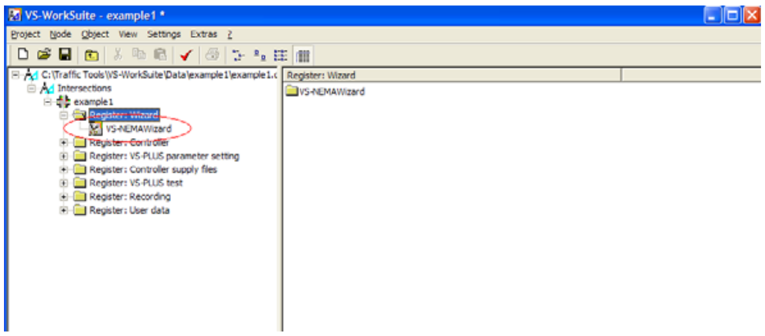


Figure 4.9 Procedure of VS-MEMA Wizard

In this example, we show how to use the VS-NEMA wizard to design the signal control parameters of a fully actuated four-leg intersection.

Create the simulation model

As shown in Screen 4.62, select "Register: Wizard", and create a new wizard for this project.

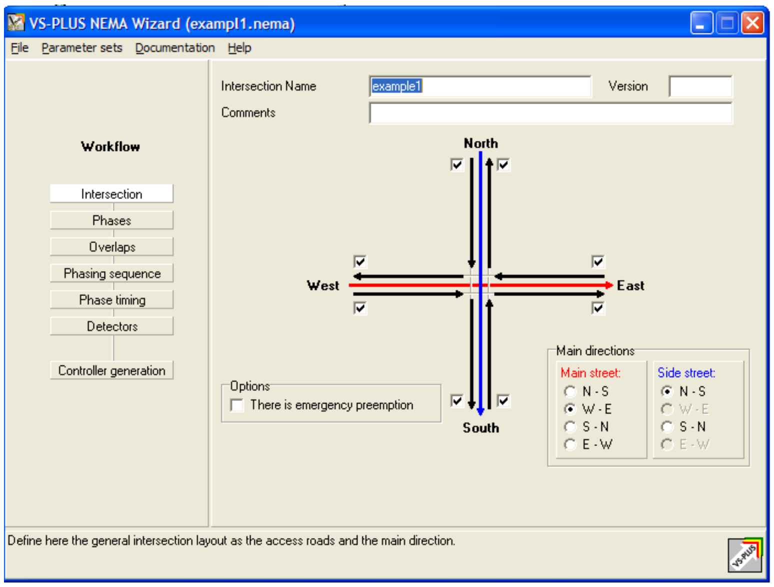

VS-WorkSuite - example1 *
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-WorkSuite\Data\example1\example1.c
Intersections
example1
Register: Wizard
VS-NEMAWizard
Register: Controller
Register: VS-PLUS parameter setting
Register: Controller supply files
Register: VS-PLUS test
Register: Recording
Register: User data
Register: Wizard
VS-NEMAWizard

Screen 4.62. Create a new VS-NEMA wizard for the intersection

In VS-NEMA Wizard, follow seven steps to design the intersection signal control.

(1) Step 1: Intersection Layout

As shown in Screen 4.63, define the intersection layout as the side street (minor roadway) and the main street (major roadway).



Intersection Name: example1
Version:
Comments:

Options
☐ There is emergency preemption

Main directions

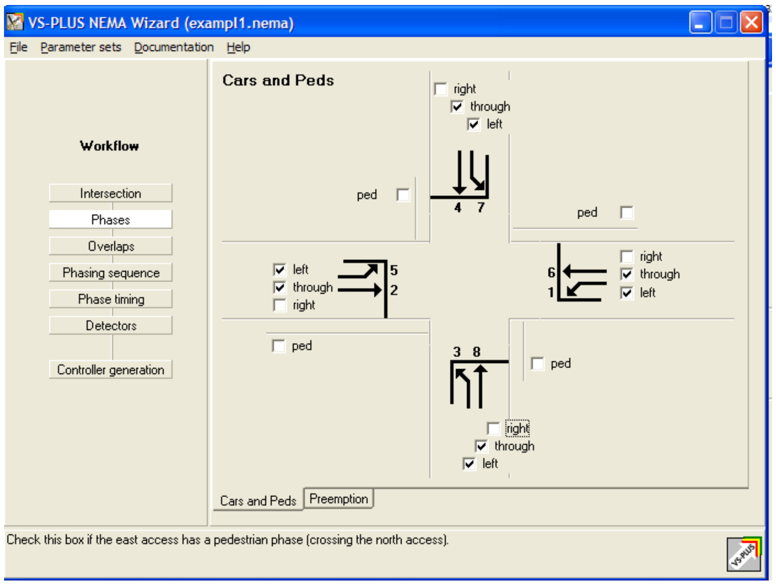
Main street:	Side street:
○ N - S	● N - S
● W - E	○ W - E
○ S - N	○ S - N
○ E - W	○ E - W

Define here the general intersection layout as the access roads and the main direction.

Screen 4.63. Define the intersection

(2) Step 2: Phases

In Screen 4.64, define the phases based on the movements of traffics at the intersection. Each approach can have a through phase, left turn and right turn phase. Through phases and left turn phases have standard NEMA numbering, separate right turn phases are numbered as overlaps. The pedestrian crossing assigned to one direction of an intersection approach is the one crossed by its right turn traffic.



North approach: ☐ right, ☑ through, ☑ left; ped ☐

East approach: ped ☐; ☐ right, ☑ through, ☑ left

West approach: ☑ left, ☑ through, ☐ right; ☐ ped

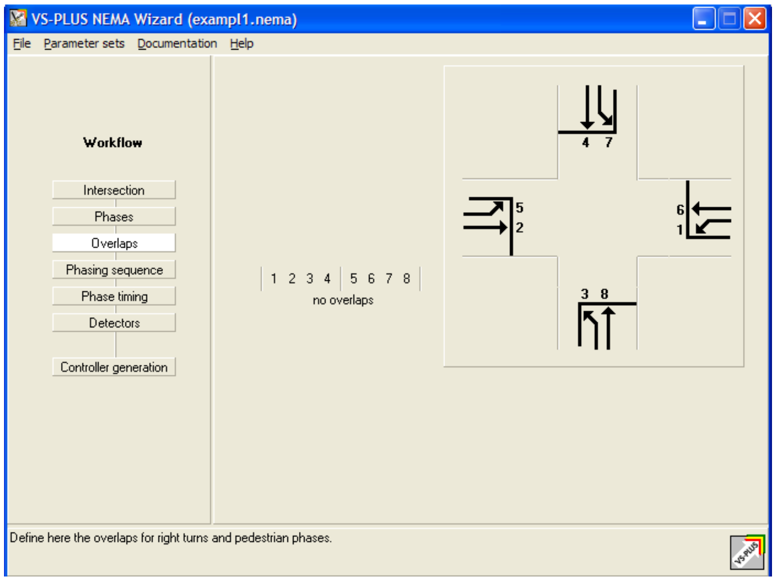
South approach: ☐ ped; ☐ right, ☑ through, ☑ left

Check this box if the east access has a pedestrian phase (crossing the north access).

Screen 4.64. Define the phases

(3) Step 3: Overlap

Define all of the possible overlaps (not needed in this example).

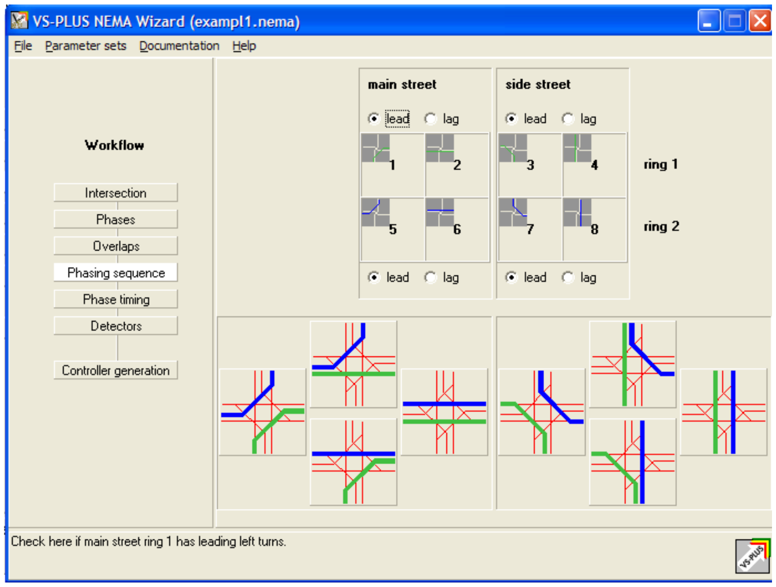


Define here the overlaps for right turns and pedestrian phases.

Screen 4.65. Define the overlap

(4) Step 4: Phasing Sequence

Once the phases and overlaps are defined, the phasing sequence can be shown in Screen 4.66. User can choose the desired phase patterns. Each ring can be set to lead or lag.



main street: ring 1 [✓ lead] lag; ring 2 [✓ lead] lag

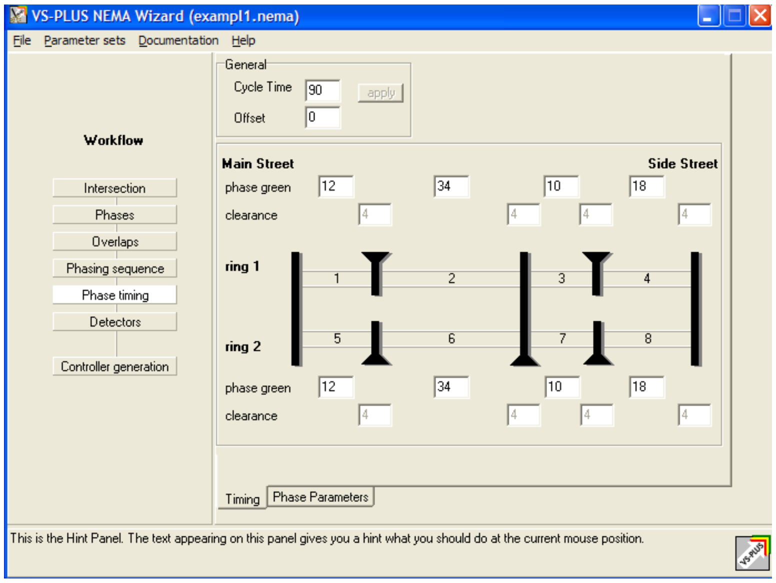
side street: ring 1 [✓ lead] lag; ring 2 [✓ lead] lag

Check here if main street ring 1 has leading left turns.

Screen 4.66. Define the phase sequence

(5) Step 5: Phase Timing

Through Screen 4.67, it defines cycle time, offset, green time, and clearance time. Here, the phase green will be used as the maximum green time 1 for the phase.



General

Cycle Time 90

Offset 0

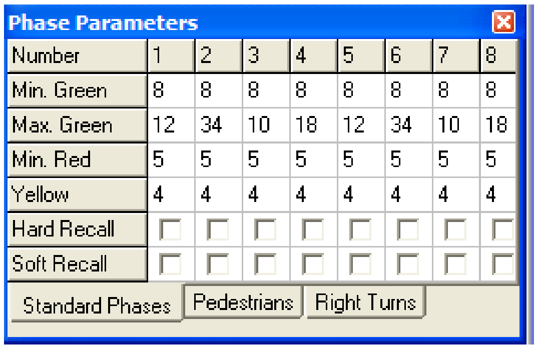
Main Street — Side Street

phase green	12	34	10	18
clearance	4	4	4	4
ring 1	1	2	3	4
ring 2	5	6	7	8
phase green	12	34	10	18
clearance	4	4	4	4

This is the Hint Panel. The text appearing on this panel gives you a hint what you should do at the current mouse position.

Screen 4.67.   **Define the phase timing**

As shown in Screen 4.68, a sub-panel "Phase Parameters" is used to define all basic parameters (Min. green, Max. green, etc.)

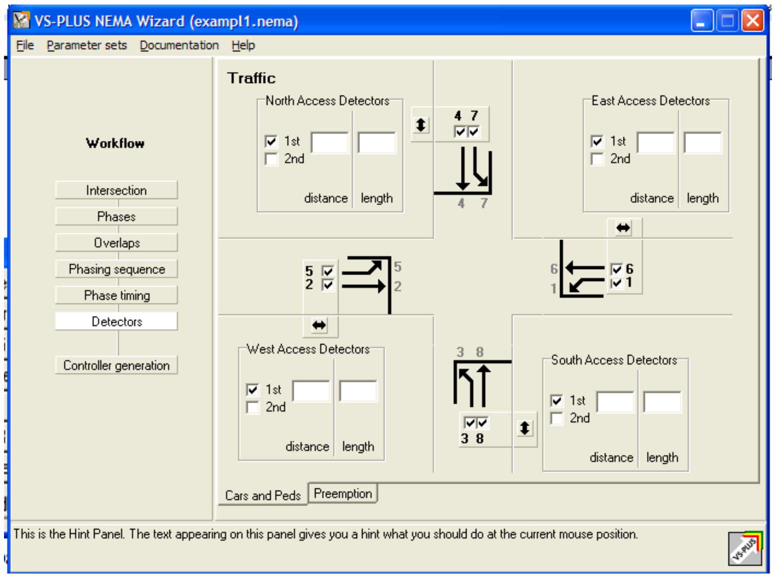


Number	1	2	3	4	5	6	7	8
Min. Green	8	8	8	8	8	8	8	8
Max. Green	12	34	10	18	12	34	10	18
Min. Red	5	5	5	5	5	5	5	5
Yellow	4	4	4	4	4	4	4	4
Hard Recall	☐	☐	☐	☐	☐	☐	☐	☐
Soft Recall	☐	☐	☐	☐	☐	☐	☐	☐

Screen 4.68.   **Define phase parameters**

(6) Step 6: Detectors Setting

In Screen 4.69, it defines detectors for the corresponding approach. In the current version, the detector position and length are only needed for informational purposes and are not being used in VS-PLUS.

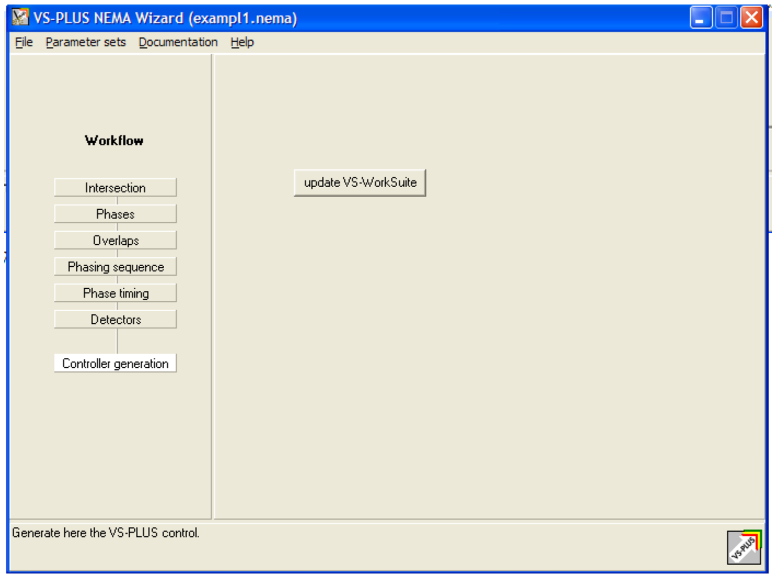


This is the Hint Panel. The text appearing on this panel gives you a hint what you should do at the current mouse position.

Screen 4.69. Define the detectors

(7) Step 7: Controller Generation

Update VS-WorkSuite, and generate the VS-PLUS control parameters.



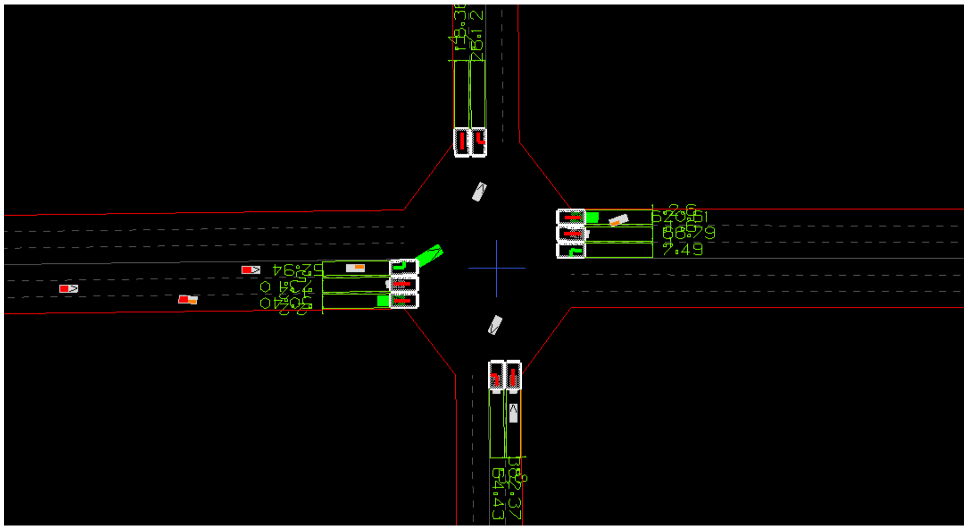
Generate here the VS-PLUS control.

Screen 4.70. Generate the VS-PLUS parameters

After setting the signal timing parameters in the Wizard, the NEMA Wizard generates corresponding VS-PLUS parameters according to NEMA standards.

Evaluation

In the current version, VS-NEMA Wizard generates two signal control plans with two corresponding sets of phase times and phase flag parameters. One of these plans is used for free operation (without permissive period plan), and the other is coordinated operation (with permissive period plan). The default in VS-PLUS is set for free operation. Since the sample intersection runs free operation, there is no frame signal plan (permissive period plan).

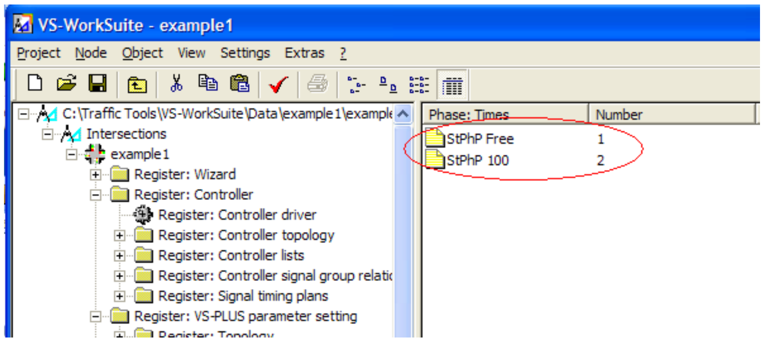


Screen 4.71. Simulation runs for isolated fully actuated intersection

It is worth of note that several parts of the wizard need to be improved or corrected to avoid the risk for generating incorrect VS-PLUS parameters. Some common errors in the current version are shown below.

(1) Incorrect parameter generation in group "Parameter set: Definition"

In Screen 4.72, "StPhP Free" contains the phase times used for free operation, while the screen titled "stPhP 100" is used for phase times in coordinated operation.

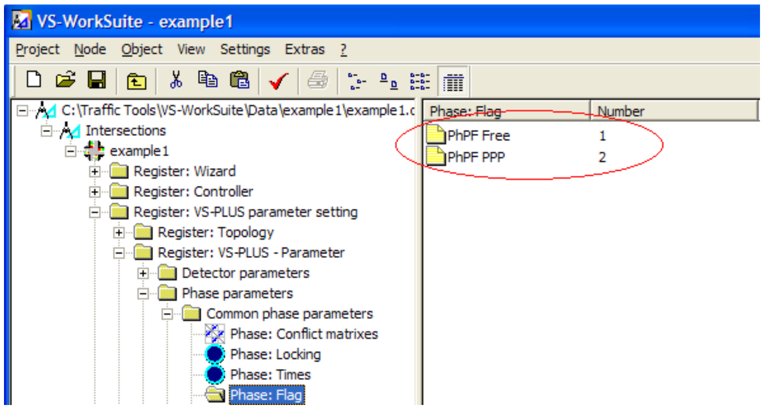


VS-WorkSuite - example1
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-WorkSuite\Data\example1\example
Intersections
example1
Register: Wizard
Register: Controller
Register: Controller driver
Register: Controller topology
Register: Controller lists
Register: Controller signal group relatio
Register: Signal timing plans
Register: VS-PLUS parameter setting
Register: Topology

Phase: Times	Number
StPhP Free	1
StPhP 100	2

Screen 4.72. Phase Times

In Screen 4.73, “PhPF Free” contains the phase flags for free operation, and “PhPF 100” shows the phase flags for coordinated operation.


VS-WorkSuite - example1

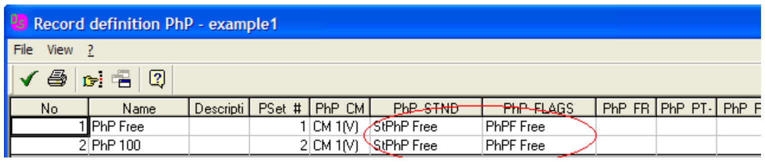
Project Node Object View Settings Extras ?

- C:\Traffic Tools\VS-WorkSuite\Data\example1\example1.c
  - Intersections
    - example1
      - Register: Wizard
      - Register: Controller
      - Register: VS-PLUS parameter setting
        - Register: Topology
        - Register: VS-PLUS - Parameter
          - Detector parameters
          - Phase parameters
            - Common phase parameters
              - Phase: Conflict matrixes
              - Phase: Locking
              - Phase: Times
              - Phase: Flag

Phase: Flag	Number
PhPF Free	1
PhPF PPP	2

Screen 4.73. Phase flag

In Screen 4.74, in the phase definition, both sets of phase parameters use the parameters for free operation. The second set of phase parameters should use those for coordinated operation.


Record definition PhP - example1

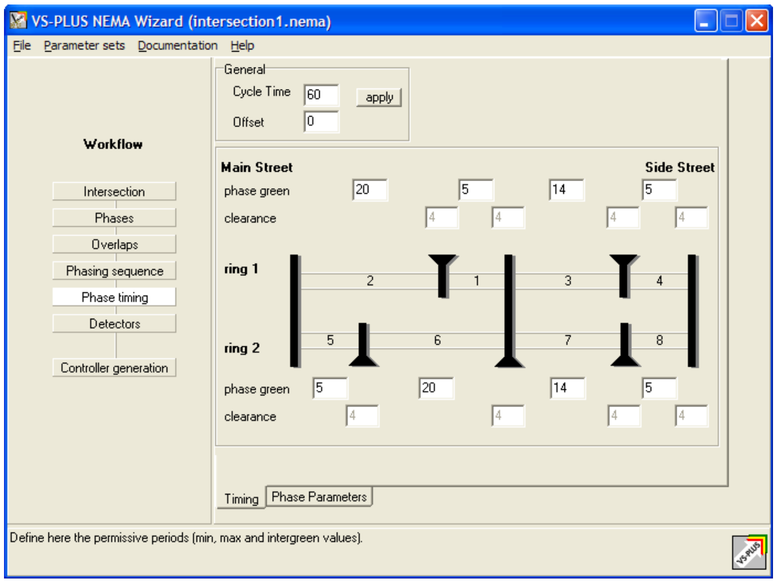
File View ?

No	Name	Descripti	PSet #	PhP CM	PhP STND	PhP FLAGS	PhP FR	PhP PT-	PhP F
1	PhP Free		1	CM 1(V)	StPhP Free	PhPF Free			
2	PhP 100		2	CM 1(V)	StPhP Free	PhPF Free			

Screen 4.74. Parameter set definition Phase 2

(2) Incorrect parameter generation in group “Phase Times”

In some cases, the phase timer parameters (maximum green, minimum green) are not correctly generated. In these cases, users must manually adjust the parameters. As shown in Screen 4.75 and Screen 4.76, errors can be found in the phase times; the maximum green time 1 for phase 1 and 2 as well as the minimum green time 2 for phase 5 and 7 are incorrect and must be manually changed.



General

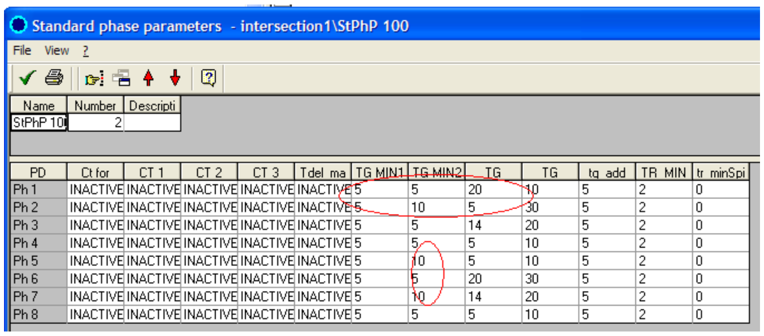
Cycle Time 60

Offset 0

	Main Street			Side Street
ring 1 phase green	20	5	14	5
ring 1 clearance	4	4	4	4
ring 2 phase green	5	20	14	5
ring 2 clearance	4	4	4	4

Define here the permissive periods (min, max and intergreen values).

Screen 4.75. Lead-leg signal timing plan



Name	Number	Descripti
StPhP 10	2	

PD	Ct for	CT 1	CT 2	CT 3	Tdel ma	TG MIN1	TG MIN2	TG	TG	tq add	TR MIN	tr minSpi
Ph 1	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	20	10	5	2	0
Ph 2	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	10	5	30	5	2	0
Ph 3	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	14	20	5	2	0
Ph 4	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	5	10	5	2	0
Ph 5	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	10	5	10	5	2	0
Ph 6	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	20	30	5	2	0
Ph 7	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	10	14	20	5	2	0
Ph 8	INACTIVE	INACTIVE	INACTIVE	INACTIVE	INACTIVE	5	5	5	10	5	2	0

Screen 4.76. Phase Times

(3) Detectors setting need to be improved

As shown in Screen 4.69, the detector position and length are only needed for informational purposes in the current version VS-NEMA. The second detector currently uses the beta version and is not yet used in VS-NEMA.

(4) Conversion of the Different Control running manners

The default control type of VS-PLUS is free operation. The user must manually change to the coordinated operation if it is desired.

4.2.4 Three Fully Actuated Coordinated Intersections

Coordination is a mode of signal operation designed to enable an uninterrupted flow of traffic through a series of traffic signals. If traffic volumes are high and signals are closely located, coordination is necessary to avoid excessive delays and stops. There are several types of control logic used in coordination. The most commonly used is found in the closed-loop system. Figure 3.4 is the phase diagram of the actuated signal coordination. There are two critical functions in the closed-loop system for signal coordination.

(1) Provide synchronization of the coordinated phases.

Yield points are used to synchronize the coordinated phases and maintain the background cycle length. All coordinated intersections have the same system clock reference point, which is usually the start point of the signal coordination plan. For actuated signal coordination, the coordinated phase of every coordinated intersection has a fixed series of yield points, and the difference between the yield points is the background cycle length. The yield points are also local clock reference points used for the remaining non-coordinated phases.

(2) Maintain the minimal bandwidth for the coordinated phases.

In order to maintain the minimal bandwidth, all non-coordinated phases have to be cut at certain points. These so-called force-off points are usually referenced to the local clock reference point (yield point).

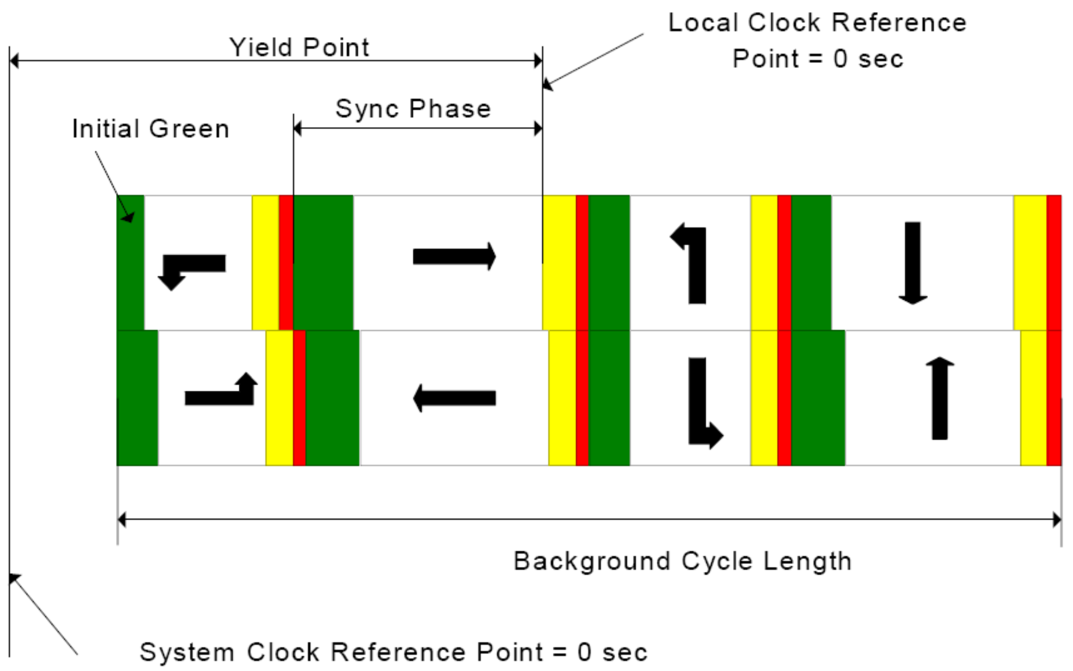

Local Clock Reference Point = 0 sec
Yield Point
Sync Phase
Initial Green
Background Cycle Length
System Clock Reference Point = 0 sec

Figure 3.4 Actuated signal coordination (6)

The closed-loop system is not used in the VS-PLUS simulation software. Instead, coordination in VS-PLUS is achieved by the use of permissive period plans. As shown in Figure 3.5, the two phases are coordinated if the permissive signals are identical to the correctly set offset.

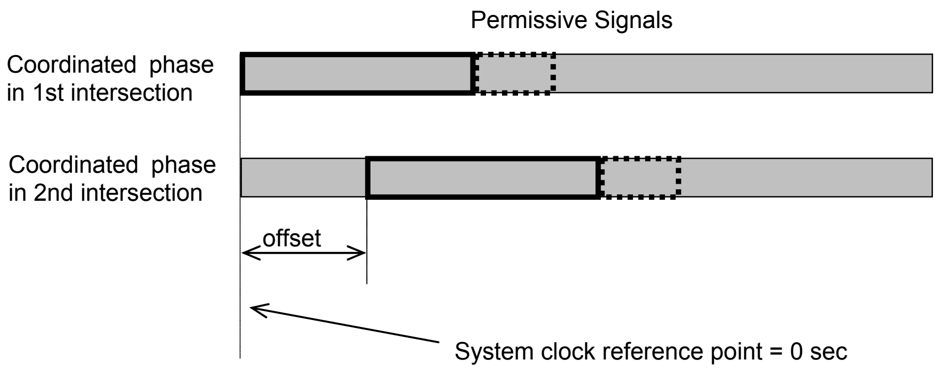
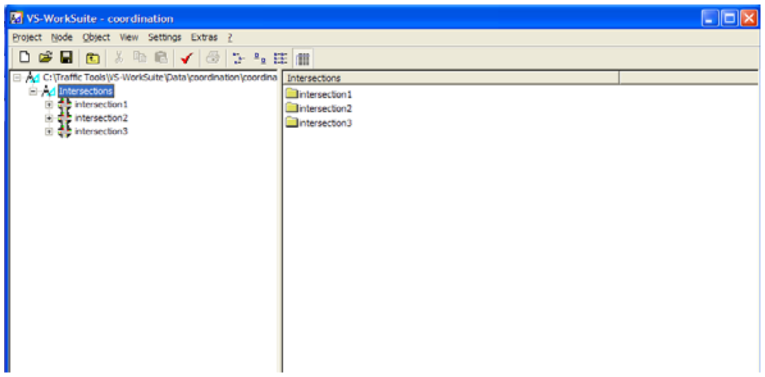

Permissive Signals
Coordinated phase in 1st intersection
Coordinated phase in 2nd intersection
offset
System clock reference point = 0 sec

Figure 3.5 Permissive signals of coordinated phases in VS-PLUS

Since the permissive period plans define the basic coordination rules, VS-PLUS evaluates the actual detector measurements and performs fine-tuning for each cycle. Through the following example, we illustrate the key procedures needed for building coordinated signals in Paramics/VS-PLUS.

Create the simulation model

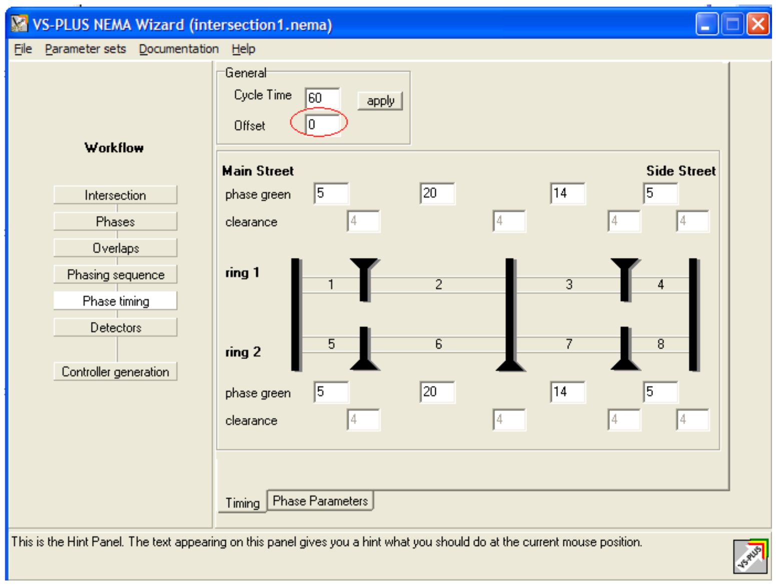
(1)Create three intersections for coordination project in VS-PLUS. (Please refer to Section 3.1.1).


VS-WorkSuite - coordination
Project Node Object View Settings Extras ?
C:\Traffic Tools\VS-WorkSuite\Data\coordination\coordina
Intersections
intersection1
intersection2
intersection3
Intersections
intersection1
intersection2
intersection3

Screen 4.77. Create a VS-PLUS project

(2) Setting VS-PLUS Parameters Using VS-NEMA Wizard

First, VS-NEMA wizard is used to generate the VS-PLUS parameters for the three intersections. For illustrative purpose, the three intersections were given same signal timing plans. The coordinated phases are phase 2 and 6. The offsets for these intersections are 0, 8 and 16 seconds. Next, manually change the control manner to coordinated operation and export all of the VS-PLUS parameters.

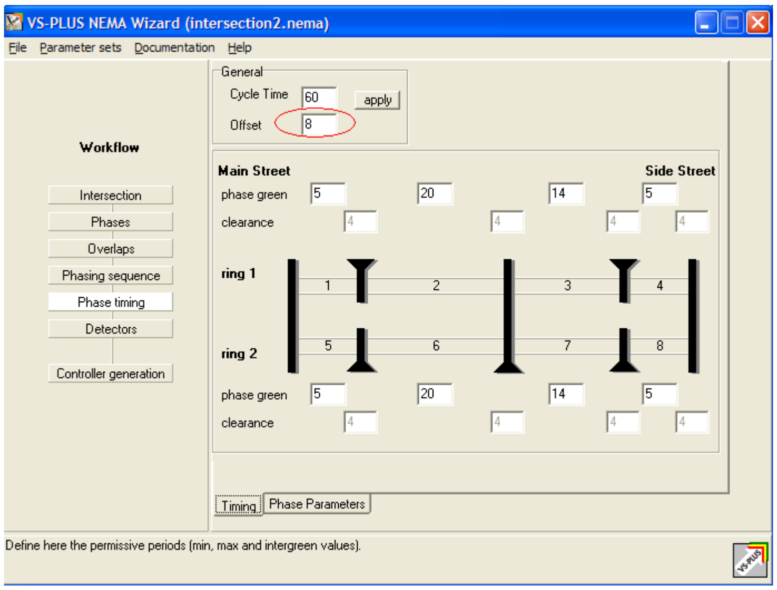


General: Cycle Time 60, Offset 0

	Main Street			Side Street
phase green	5	20	14	5
clearance	4	4	4	4
ring 1	1	2	3	4
ring 2	5	6	7	8
phase green	5	20	14	5
clearance	4	4	4	4

This is the Hint Panel. The text appearing on this panel gives you a hint what you should do at the current mouse position.

Screen 4.78. Design the first intersection

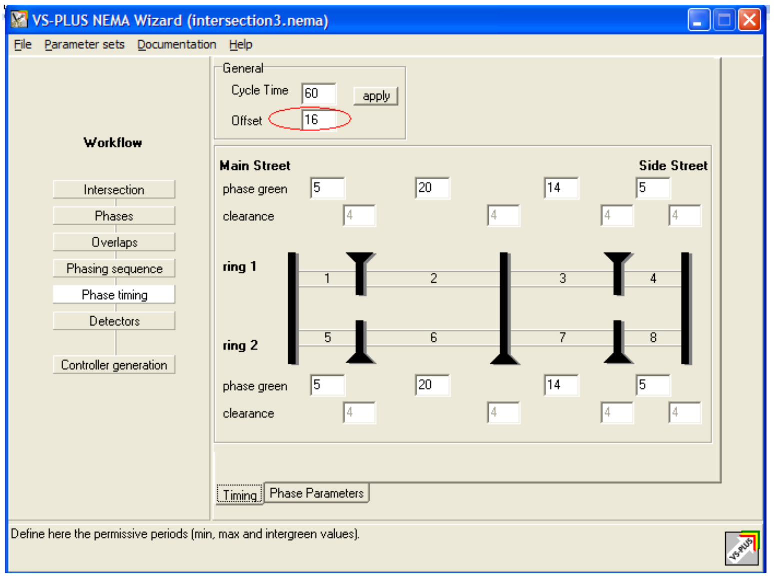


General: Cycle Time 60, Offset 8

	Main Street			Side Street
phase green	5	20	14	5
clearance	4	4	4	4
ring 1	1	2	3	4
ring 2	5	6	7	8
phase green	5	20	14	5
clearance	4	4	4	4

Define here the permissive periods (min, max and intergreen values).

Screen 4.79. Design the second intersection



VS-PLUS NEMA Wizard (intersection3.nema)

General
Cycle Time 60
Offset 16

Main Street				Side Street
phase green	5	20	14	5
clearance	4	4	4	4
ring 1	1	2	3	4
ring 2	5	6	7	8
phase green	5	20	14	5
clearance	4	4	4	4

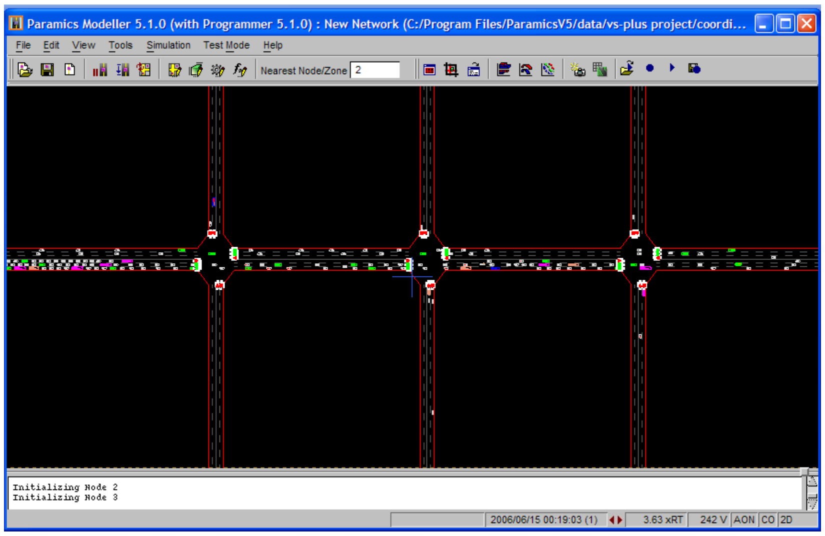
Define here the permissive periods (min, max and intergreen values).

Screen 4.80. Design the third intersection

Refer to Section 3.1, generate parameters files in VS-PLUS; create Paramics Model and set the Paramics/VS-PLUS interface.

Evaluation

Screen 4.81 shows a snapshot of Paramics/VS-PLUS model running a coordinated signal timing plan. Since VS-PLUS is controlled by parameters, coordination is relatively easy to operate through the setting of offsets. Some traffic-responsive features of VS-PLUS can also be observed in the operation of coordinated control. For example, it was observed that when excessive traffic from the side streets exists, VS-PLUS can update green time of the main direction to keep a certain bandwidth while at the same time discharges queued vehicles on the side streets whenever it finds some free time in the signal cycle.



Paramics Modeller 5.1.0 (with Programmer 5.1.0) : New Network (C:/Program Files/ParamicsV5/data/vs-plus project/coordi...

Nearest Node/Zone 2

```
Initializing Node 2
Initializing Node 3
```

2006/06/15 00:19:03 (1) 3.63 xRT 242 V AON CO 2D

Screen 4.81. Running in simulation

4.3 Conclusions about VS-PLUS

The integrated Paramics/VS-PLUS simulation is currently not available for Paramics Viewer, but it can implement most of the functions of VS-PLUS in Paramics Modular. Through the evaluation of three simulation models in this report, the integrated Paramics/VS-PLUS simulation is proven to be effective in modeling irregular intersections, isolated fully actuated signals, and actuated coordinated signals. The core part of the VS-PLUS control algorithm is the definition of the permissive signals and frame signals. The permissive period plan is the summation of all permissive signals for all phases which provides a great flexibility for the users to model most types of signal control. VS-NEMA Wizard is a user friendly tool for American user. Users are allowed to design their signal plans based on the NEMA standard in the Wizard.

Several shortcomings of the Paramics/VS-PLUS model need to be addressed. First, detailed guidelines as well as examples are needed for setting the permissive period plan. It is very difficult for users to understand the definition and application procedures of the permissive period plan. Secondly, most of the VS-PLUS parameters are complex and hard to understand by beginners as they use VS-PLUS specific definitions. Thirdly, a certain number of files in current version are still in German. This was found mostly in “Help” files.

Conclusions

With the continuing increase in urbanization and traffic growth, traffic engineers have been forced to develop new traffic control systems in order to maintain safety and improve the efficiency of existing transportation systems. This project explored several different methods of increasing accuracy and performance in traffic control system simulation, and using simulation to optimize traffic control systems. Methods examined include Hardware-in-the-Loop Simulation (HiLS), the use of plug-in traffic signal controller modules more feature-rich than the default Paramics traffic signal simulation, and the use of the VS-PLUS traffic signal controller software, which provides software emulation of the traffic signal controller which is based on the actual code running in the controller. The major findings of the research can be summarized as follows:

The Hardware-in-the Loop Simulation (HiLS) using the McCain Traffic Supply/NIATT Controller Interface Device (CID) is a useful tool for testing and evaluation of different traffic control system features and algorithms. The researchers at the Universities of Minnesota and Utah used HiLS communicating through a real-time linkage to develop a traffic flow prediction model. Researchers at California PATH have continued to use it for lab testing of improved control strategies for transit priority. A Paramics interface tool for HiLS was also developed that can be used for educational purposes to provide students with a better understanding of traffic signal controller operation and programming.

Comparisons of different control modes shows that adaptive signal control performed better during periods of high demand compared to actuated signal control and pre-timed signal control. Though the actuated signal control performed better when demand was low, the inclusion of a dynamic signal optimization model and a turn count estimation model may improve the performance of adaptive signal control at lower traffic demands.

Real-time linkage to Paramics simulations using the CID can be done with 170 and 2070 controllers. The current HiLS set up requires both a CID and a traffic controller to test signal control logic on a single intersection. For testing multiple intersections, multiple pairs of hardware would be required, which makes it impractical to use HiLS for each controller in a large traffic signal network. However, simulations using the UC Irvine Paramics Signal Controller plug-in, which is a more faithful simulation of actual traffic signal behavior than the default Paramics traffic signal control model, show behavior similar to 2070 HiLS. Developing traffic control software that includes all the advanced features available in the actual controller into an application program interface would eliminate the CID and the actual traffic controller from the current HiLS setup.

The VS-PLUS traffic signal controller software that has been widely used in Europe, has an implementation for the 2070 controller which includes a traffic signal controller emulator and Paramics integration. The integrated Paramics/VS-PLUS model provides many useful functions for emulating traffic signals in traffic simulation, especially actuated signals. A User’s guide was developed in this study for the application of the software.

APPENDIX A: Answers from VS-PLUS during evaluation

The following questions we asked during the evaluation of Paramics/VS-PLUS interface were answered by VS-PLUS developer Dr. Thomas Riedel.

Q1: I found in your presentation "Introduction for Caltrans", you combined VS_PLUS with Paramics. Could you tell me how did you deal with it? Such as, now I have a Paramics model, and I have a signal time plan by VS_PLUS. How could I combine them together? BTW, when I use VS_PLUS, one of the dialog boxes is German.

A1: you need to understand the following points: - For the simulation with Paramics, you need to define the Paramics network with the right signal id's and the right detector id's - At the same time you need to define the Paramics node to be controlled by the VS-Emulator - The VS-Emulator needs to parameter files that you have to define within Paramics as well: the basic.vcb and the vsplus.vcb - These files are generated in the VS-WorkSuite (under Export) The first time you start Paramics with VS-Emulator, you will be asked to license your copy of the VS-PLUS plugin for Paramics. Please send us the "toBeLicensed" file, and we will generate your license number.

Q2: I have several questions about VS-PLUS basic concept. Because, in VS-PLUS, some concepts are not as same as general ones. 1. In the example given in "first steps" manual, about the concept of "signal group", does this mean you group LT (left turn) and RT (right turn) TH (though) vehicles together with the same direction? 2."Controller signal group relations" in "controller topology" and "common phase problem" in "VS-PLUS parameters" in both of them, the conflict matrix is defined. Are they always the same matrix or not? 3. In "controller topology", the signal timing plan is defined. For actuated signal control, is this the initial computation for permissive signal plan? 4. For fixed time signal control, after setting the controller topology, do we need to set the other parameters? 5. I am still confusing about the concept "permissive signals". Is this similar to the "phase correspond to signal group"?

A2: Signal Group A signal group typically is a single signal head, but can also be group of signal heads that show always the same lamp combinations, as if they were hard-wired. Grouping of signal heads that could show different states is done in the VS-PLUS logic. If you hard-wire LT traffic with the opposite RT traffic, you can talk about a single signal group. Normally in VS-PLUS this is not done in this way, but by combining the two signal groups to a phase combination or by connecting them as display element (UN-conditional display elements) 2. Conflict Matrix VS-PLUS uses 2 different conflict matrices:

- Controller conflict matrix: this matrix shows the conflicts between signal groups
- VS-PLUS conflict matrix: this matrix shows the conflicts between traffic streams.

As several traffic streams can use the same lane and hence the same signal group, this matrix can only be more restrictive than the controller conflict matrix. If you have a 1:1 mapping, i.e. there is 1 traffic stream per signal head, both matrices are identical. This is the case in the mentioned example. 3. Signal Timing Plan The signal timing plan can be used as the base for generating a initial persmissive period plan (a synonym is: frame signal plan). 4. Fixed Time Control For fixed time control, only the signal timing plan has to be set. This plan is taken by the controller for the emergency program. In VS-PLUS, a fixed time control is usually generated by developing a very restrictive permissive period plan. 5. Permissive Signals A synonym is: frame signals. Please refer to chapters 2.2 and 2.3, and 3.1 to 3.3. A permissive signal tells the traffic stream

- when to consider calling by detectors or other traffic streams
- when to consider green extension.

It gives the frame for VS-PLUS in order to decide, when a traffic stream can be served. After this decision, VS-PLUS breaks the traffic streams down to the phases and signal heads.

Q3: I am working on code a very simple example using VS-PLUS. And building the Paramics model shows the result. I use the VS-NEMA Wizard to build a four leg intersection and generate the VS-PLUS parameters.And then export the vcb files. In Paramics, I build a four leg intersection model and define the detectors and external signal. But the signal doesn't work right. Could you help me find out the problem?

A3: some questions and some hints: - are there error messages in the log window when you start the simulation run? - does the VS-Emulator start when you are loading the network? - did you give the same node number everywhere, i.e. in   - the Paramics network (TC Node ID)   - in the Wizard parameters?   I suppose you want to use "5" as the node number / intersection number - the detector names must be numerical or numerical after the dot, e.g   "1" or "node5.1"; this numerical value is the same you are using in the   detector configuration in the Wizard window

Q4: I used VS-PLUS NEMA Wizard build a four leg intersection and generated the VS-PLUS parameters. But I don't understand the permissive period plan which generated by NEMA wizard. Could you explain how to set the permissive period plan for this example? And I try to use help file. For some reason, almost all help files are German version. So, I am wondering whether you could send me the English version help files.

A4: Unfortunately there is no English online help for VS-WorkSuite, there is only the manual. Do you have it? It should be part of the software delivery on the CD. If you don't have it, I will send you a pdf copy. In this manual there are chapters about the Permissive Period Plan and, at least as important, about the pointers for major and minor phases. VS-PLUS works perfectly without a PPP - the reason for the PPP is to give a frame to the VS-PLUS optimizations in order to guarantee cycle time and an eventual coordination. The sequence of phases is defined by the pointers; the timing is restricted by the PPP - that's maybe the simple way to explain. You might be wondering why the PPP does not look like the signal plan - that's why. You have to check in parallel the pointers that guarantee that the first phases are served first, even though the PPP would allow them.

Q5: I have already finished the Paramics example using VS-PLUS to control the external signal. It works very well. The next step, I want to coordinate several intersections together. In VS-PLUS, do I need build intersections separately or I can build them into one model? Where I can define the master controller? What parameters I need to define?

A5: If you want to coordinate several intersections, you should set the offset to what you think the coordination offset should be. You need to build them separately because the offset is a parameter within VS-PLUS. There is no master controller needed, as the offsets are fixed in the present configuration. In order to safe efforts, you can import an existing NEMA configuration into the one you are working on, and then simply modify the offset and export the parameters to VS-WorkSuite

Q6: I have already finished the coordination example using the method as you mentioned in the last email. It looks good. But I have some questions about the parameters setting. For coordination, in VS-PLUS, are there any parameters can define the FORCEOFF for each signal group? For 170 or NEMA controller, Setting FORCEOFF is the way we can assure the coordination bandwidth. Generally, FORCEOFF is the parameter which defines, at which local time point, the green should be turned off because of the requirement by coordination. VS-PLUS did the perfect job for gap out and maxgreenoff. I don't know whether it can perform FORCEOFF or where I can define the parameters for FORCEOFF. (Does it define in the permissive signal in VS-PLUS?)

A6: FORCEOFF: If I got it right, you want to close a traffic stream at certain moments. You can do this with the permissive signal, having set the correct red flag: - Set the frame end at the point of time where you want to force off the traffic stream, - and set the red flag of this traffic stream to "end of frame" (or similar, I don't recall   the English term exactly). This makes the traffic stream force off always.

Q7: Another question is for detector setting. So far, the detectors I used in the Paramics model are long loops (6m*24m). The long loop performs the impulse with occupancy function. Can we perform the other detectors arrangement in Paramics? Such as using advanced loop(impulse) and long loop (occupancy) This kind of detectors arrangement also are showed in the manual, (page5) 3.2 Detector Evaluation.

A7: You can use any kind of detector layout in Paramics. VS-PLUS is using the rising and the falling slopes generated by the cars driving over the detectors. Just change the shape of the detectors in Paramics, and it should work

Q8: The Paramics example "sr41-trimed", the detectors for the two lanes (ex. 2 through or 2 left turn) are defined such as: "214.a.8; 214.b.8". They are according to the same signal group and same movement. How can I define them in the VS-PLUS? Do I need define the VS-PLUS detectors named as "a.8" and "b.8"?

A8: This is a naming convention in Paramics:

- The digits until the first point define the node the detectors belong to: node 214,
- The digits after the last point define the detector channel number: both channel number 8
- You are free what you want to write between the points

Conclusion: the name in VS-PLUS is free, the channel number is 8 in both cases here

Q9: If last phase has gap-off, which means in that phase, it saved some green time. There are two different ways to assign the saved green time within the signal cycle. One is that assign the saved green time to the next phase. Another is that assign all saved green time within one cycle to the last phase. Either way, the signal cycle time always keeps same. In VS-PLUS, does the signal cycle time keep same? Or it will change based on the detector call?

A9: If you have a Frame Signal Plan (or Permissive Period Plan PPP), VS-PLUS keeps the cycle time. If not, VS-PLUS runs in free mode. As the phase start and end points can move with a cycle according to the detector inputs, it might appear that the cycle is longer or shorter in some cycles. In fact, VS-PLUS counts the cycle second (TX) until it reaches the cycle time (TC, sometimes called TU). TC is constant for a program.

- If you save some green time by gapping off a phase, the next phase being servable (according to the PPP and/or the pointers to the major and minor phases) will be served.
- If this phase, being served earlier, can be gapped off earlier, the next phase will be considered. The timeout of the phase can be chosen in such a way that the saved time is kind of propagated to the next phase.
- Conclusion: the saved time is always given to the next servable phase, which can propagate it further if the parameters are chose accordingly.

Q10: In manual "understanding VS-PLUS", it mentioned that :" In fully traffic actuated controls the permissive signal runs throughout the entire cycle time for each phase. There is not split into call and extension section" I don't understand why? Does that mean in fully actuated control, every phase can be called in any time? And the green for each phase can be happened at any time in cycle? At this situation, the cycle time doesn't work?

A10: In fully traffic actuated control, the phase sequence is given by the major and minor phases. Each time a major phase is a candidate for next green, and it has been called, it is given green together with its minor phases that have calls. The length of green is given by detection parameters (gap time) and other timeout parameters. Please use the "green battle" image:

- each red phase is fighting for green
- each green phase is fighting for staying green

The fight is decided according to good or even better "arguments" for green. The permissive period plan is only one of the arguments, the major and minor series are another argument, and the detector and traffic stream parameters are "arguments" as well.

Q11: In VS-PLUS, if we want to use Semi-actuated signal control and fixed time control. How we deal with it? Is that we limit the permissive signal for fixed phase to the fixed signal time?

A11: We can restrict the permissive signal to pure cyclic behavior in not giving any optimization freedom at all to VS-PLUS. We can leave out the permissive signal. So the green is only determined according to "who wins the battle for green next"

Q12: When we use the VS-NEMA to generate the parameters, the detector's parameters didn't transfer to VS-PLUS. For example, we set the distance of detectors 20 meters in VS-NEMA, after generated parameters to VS-PLUS. The detector parameters in VS-PLUS are all 1 meters for each detector. And in VS-NEMA, we set the second detector for each phase; it did show in VS-PLUS. The attachment is the snapshot for VS-NEMA setting and VS-PLUS In Paramics, if we have a phase with two detectors. For example, detector 5 and detector 25 for phase 5. How can we set the movement ID. Is that from 2 to 5 movement ID 5 with 2 detectors: detector "3.5", detector "3.25"

A12: The detector distance is not used in the VS-NEMA Wizard right now. It is only for information purposes. The second detector is a beta version only because we did not decide yet how it should be handled by the NEMA Wizard algorithm. If you want to have both detectors uses, you must map them to one single input channel, e.g. 3.a.5 and 3.b.5. I don't know how fit you are in NEMA. If we are able to define together, maybe also with the Caltrans people, how to handle the 2nd detector in an approach, I will implement it in the VS-NEMA Wizard.

Q13: About detector waiting time (twdet) and phase waiting time (twvs) "phase waiting time corresponds to the waiting time for the phase since the first call" Here "the first call" means the first corresponding detector call or the first phase call?

A13: The first call is the first detector call. With the first detector call also the phase waiting time starts

Q14: "In order to call a phase the delay time (detector waiting time greater than comparative value delay time) must be exceeded first. Second, the permissive signal to call the corresponding phase must be set to green" There are two conditions; the first one is that the permissive signal must be in the call range. The second condition is: Compare the detector waiting time with the comparative value delay time. What is the delay time? Is that hold time for the detector?

A14: No. With the delay time you can delay the detector call. You can delay the physical call of a detector before VS-PLUS will accept it for the treatment. In Europe we have sometimes detectors far form the intersection, so with this parameter we can delay the call. The delay time parameter is used on the level of the detector treatment. The permissive signal is used on the level of the phase's treatment.

Q15: What the results from the detector evaluation? Is there a maximum detector waiting time?

A15: The result of the detector evaluation is the detector waiting time. There is no maximum waiting time

Q16: For "requ1": why it shows "permissive period end"? Does that mean it is the end point of permissive period for this phase?

A16: prep and requ should always have the same value. Prep is used for PT priority issues in order to give the frame a PT advance calling region:

- requ is the calling
- ext is the point from when green can only be extended
- end is when the frame ends.

Q17: For extension: In page 2 of manual: "Calls are dealt with in the call range and forward to activation commands whereas a phase must have green in the extension range and may only extend." So my understanding is that: the call range of permissive period defines time period to consider the call from this phase. This means the possible time period for the phase switch point (the point when the green will start). The extension range means the time period to consider green extension.

For example from the above snapshot, for phase L1:

Permissive period start: 34

Call range from 34 to 72

Extension range from 72 to 85

So the possible green start point for phase L1 is 34 to 72 if there is detector call or the switch command from the other conflicting phase. Question is that if the green starts at the call range (for example time point 40), after minimum green time 2 (for example minimum green time 2 is 8 seconds, so now the time point in cycle is 48) , the phase should consider whether need green extension or not. But now it is not in the extension range (72-85), how it can work?

Is there any problem for my understanding?

A17: Extension is always possible also in the call range – extension range means that only extension is possible.

Q18: In manual page14, the description of weak coordination and medium coordination is same. And in page 32, there are two type coordination flags: hard coordination and soft coordination. Are they corresponding to the previous weak, medium and strong coordination?

A18: Weak and hard coordination are descriptive terms only. The flags you mention have the following meaning:

There are 3 flags for the coordination.

You can set these flags:

Globally, that means that the flag is value for all phases, or

Locally, where you set the flag for each phase.

The flags have 2 states, hard or soft. The combination of the 3 flags gives the coordination types weak, medium and strong.

	Coordination flag		
	Main phase	Minor phase WITH	Minor phase WITHOUT
Weak	hard	soft	soft
Medium	hard	hard	soft
strong	hard	hard	hard

Q19: In the VS-NEMA wizard, the offset will lead to shift the permissive period signal with setting value. What other effects by the offset? There is offset matrix change, start and end in VS-PLUS. Is there any relationship between these and the offset in VS-NEMA?

A19: The offset matrices define only constant starting and ending offsets between phases.

They have no relation to the network offset that effects the permissive period plan.

The name could be confusing. Could you think about a better name for the starting and ending offsets?

Q20:

tPON: point of time for program switch on

tPOFF: point of time for program switch off

tPCHG: point of time for best program change

How to set these pointers? Actually, in my program, these pointers seem that they don't affect the running of the controller in Paramics.

A20:This is correct: these pointers are needed for real controllers in order to know when to switch the program off and on and when it is possible to switch between programs. In Paramics, this behavior is not simulated.

How to set them? Define them in the allowed region (the blue frames in the 4th line). An allowed region is a stable region without transitions and without minimum time constraints.

Q21:In the title of the table, what is TK, TFS TSF? All description tags are German.

A21:

TK: partial node

tFS: time from free to stopped (i.e. yellow time)

tSF: time from stopped to free (i.e. red-yellow time in some countries)

Beschreibung: description

Q22: When we define position, distance, lane of detectors, do they have effects on Paramics?

Or they just have effect on real controller?

A22: These parameters are only informative; they are for future use in the controller.

Q23: For phase parameters such as: Forward call Phase: Parallel extension Phase: Lead Phase: Lag   For phase lead and lag, are they related to the VS-NEMA wizard setting about lead-lag?

A23: No, this "lead" and "lag" has a different meaning to the NEMA terminology.

Q24: What is the definition of "main phase" and "minor phase"?

My understanding is that:

In Picture development, each priority element has a corresponding main phases, these main phases sequence is selected by the pre-determined criteria (depending on the signal sequence)

Minor phases are selected for main phase based on whether they are compatible with main phase.

Minor phases can be selected as main phases

In picture development, first select the next served main phase from all called phases; second select the minor phase for the selected main phase.

Is that right for my understanding?

A24: I think that you got the principle right:

In a first step, a main phase (i.e. main traffic stream) is chosen for each priority element with a calling traffic stream. When the main phase has been designated, the minor phases are chosen in a second step from the list of the minor phases.

This procedure works only when the traffic streams are calling. A traffic stream can be called not only by a detector, but also by a green flag e.g.

Q25: “Main pointer” and “permissive signal pointer”

For each phase, we define the permissive signal pointer for it.

Is main pointer defined by the permissive signal pointer of all the main phases in the signal cycle?

A25: No – theses VS-PLUS elements are very different:

The main pointer is set by the VS-PLUS algorithm during run time and it indicates the next phase to be served.

The Permissive signal pointer is a guidance element for the main pointer. Please don't touch the permissive signal pointer and leave it at the beginning of the calling area of the frame signals. The permissive signal pointer is a tool for very advanced VS-PLUS users (even not yet for me…)

Q26: Here in real-world, coordination usually use closed loop system. It has different control algorithm. Use yield point for the coordinated phase to control the local controller reference point, and forceoff for other phase to make sure the bandwidth (green time) for coordinated phase.

A26: The nice feature about VS-PLUS is that in most cases a closed loop system is not needed! As the permissive periods define the basic coordination rules, VS-PLUS evaluates the actual detector measurements and does the fine-tuning for each cycle. - Sometimes there is more inflow from the side streets of the upstream intersections, so VS-PLUS can advance green for the main direction in order to keep the flow of the main direction fluid - Sometimes the upstream green phase is not used completely, so VS-PLUS can stop the main direction and assign a bit more green time to the side street phases. The fact that no closed loop system is needed makes the system more stable, and maybe even more optimum because of the local fine-adjustment and it saves money. The full power of VS-PLUS shows when public transport has to be integrated within the coordination. In such cases, the existing US closed loop systems often decrease car traffic capacity much more than VS-PLUS does when a PT vehicle asks for its priority. Only in case of bigger changes in green time distribution, VS-PLUS has to be re-adjusted by a central algorithm. We do this in the US with PB Farradyne's OPAC algorithm.

Q27: In fact, I feel confuse about "coordination" in VS-PLUS. In manual, there are two sites about coordination page13 3.5.3 synchronization with permissive period plan (coordination) page 32 8.4 coordination flag, In these two parts, the main point is about "how strong the synchronization of the green commands with the framework signal plan is"

A27: You are right. co-ordination is used in two different ways in the manual. There is the co-ordination in the normal sense, co-ordination between intersections. Page 13 refers to this co-ordination, and this co-ordination is obtained by the permissive period plan. The result is that the green and red phases have to follow certain timing rules, which results in a co-ordination for traffic. The co-ordination flag mentioned on page 32 has nothing to do with this co-ordination. It defines how strongly the permissive periods have to be respected by mainly the green commands, i.e. if the permissive periods have to be followed exactly or if there is a certain degree of tolerance for green without permissive period, in certain cases. Could you think about a better name for the co-ordination flag?

Q28: How to set the permissive signal plan for the coordinated intersections? For example: Usually, the signal cycle length for the coordinated intersections is same. I use VS-NEMA to generate three coordinated intersections. Set the offsets for them. In the frame signal plans for coordinated actuated signals of two adjacent intersections, For permissive signals, both coordinated signals have same setting in adjacent intersections The delay between them is offset. Is that right for my setting?

A28: You are perfectly right with the NEMA Wizard handling for co-ordination of adjacent intersections.

Q29: I think phase timer should use the second one.

Is that a problem for VS-NEMA?

A29: Good observation!! – This was an error in the NEMA-Wizard. I will correct it as soon as possible and send you an updated version.

References

Bertini, R, Lindgren, R., Tantiayanugulchai, S. (2002) Application of PARAMICS Simulation At a Diamond Interchange. Portland State University Research Report: PSU-CE-TRG-02-02.

Bullock, D. and Catarella, A. (1998) A Realtime Simulation Environment for Evaluating Traffic Signal Systems*Transportation Research Record,* 1634**,** 130-135.

Chang, G.L. and Tao, X. (1997) Estimation of Time-Dependent Turning Fractions at Signalized Intersections. *Transportation Research Record 1644, pp. 142-149*.

Chu, L., Liu, X. and Oh, J. S. (2004) A Calibration Procedure for Microscopic Traffic Simulation, In *TRB Annual Meeting*.

Chu, L., Liu, H., Smolke, B. and Recker, W. (2003). *PARAMICS Plugin Document – Actuated Signal Control,* PATH ATMS Center, University of California, Irvine, Irvine, CA.

Davis, G.A. and Lan, C.J. (1995) Estimating Intersection Turning Movement Proportions from Less-Than-Complete Sets of Traffic Counts. *Transportation Research Record 1510, pp. 53-59*.

Dickey, S. R. Bougler, B., Kretz, P, Nelson, D, Ko, J. and Misener, J.A. (2008) Intersection Signal Controllers and Vehicle-Infrastructure Integration: Putting Safe Intersections to Practice,Transportation Research Board 2008 Annual Meeting, TRB Paper 08-2627

Duncan, G. I. (1995) PARAMICS wide area micro-simulation of ATT and traffic management, In *Proceedings of 28th International symposium on Automotive Technology and automation (ISATA)*, Stuggartt, Germany, pp. 475-484.

Engelbrecht, R., Poe, C. and Balke, K. (2001) Development of a Distributed Hardware-in-the-Loop Simulation System for Transportation networks, In *Transportation Research Board Annual Meeting, National Research Council*, Vol. Preprint # 990599, Washington, D.C.

Gartner, N. H. (1990) OPAC, In *Control, Computers, Communications in Transportation: Selected Papers from the IFAC Symposium*, New York, pp. 241-244.

Gartner, N. H., Stamatiadis, C. and Tarnoff, P. J. (1995) Development of advanced traffic signal control stratergies for IVHS: A multi-level design*Transportation Research Record*, 1494.

Head, K. L. (1995) An Event-Based Short-Term Flow Prediction Model*Transportation Research Record,* 1510**,** 45-52.

Head, K. L., Mirchandani, P. B. and Sheppard, D. (1992) Hierarchical Framework for Real-Time Traffic Control*Transportation Research Record*, 1340**,** 82-88.

Henry, J. J. and Farges, J. L. (1989) PRODYN, In *Proc., 6th IFAC-IFIP-FORS Symposium on Transportation*, Oxford, England, pp. 505-507.

Highway Capacity Manual, (2000) Transportation Research Board, National Research Council, Washington, D.C.

Hunt, P. B., Robertson, D. I., Bretherton, R. D. and Royle, M. C. (1982) The SCOOT Online Traffic Signal Optimization Technique*Traffic engineering & Control*, **23,** 190-192.

Kreer, J. B. (1975) A Comparison of Predictor Algorithms for Computerized Traffic Control Systems. *Traffic Engineering, Vol. 45, No. 4, pp. 51-56*.

Kyte, M., Abdel-Rahim, A. and Lines, M. (2003) Traffic Signal Operations Education through Hands-On-Experiences: Lessons Learned from a Workshop Prototype., In *82nd Annual Meeting of Transportation Research Board*, Washington D.C.

Kyte, M., Bullock, D. and Abdel-Rahim, A. (2001) McCain-NIATT Controller Interface Device (CID II) Online User Manual, Version 2.0.

Lowrie, P. R. (1992) SCATS: A Traffic Responsive Method of Controlling Urban Traffic Control. *Roads and Traffic Authority*.

Lee, D., Chandrasekar, P. and Cheu, R.-L. (2001) Customized Simulation Modeling Using Paramics Application Programmer Interface, In *IEEE Intelligent Transportation System Conference*, Oakland, CA, USA.

Lin, W., Kulkarni, A. and Mirchandani, P. (2003) Arterial Travel Time Estimation for Advanced Traveller Information Systems. CD-ROM, 82nd Transportation Research Board Annual Meeting, Washington D. C.

Liu, H. X., Chu, L., and Recker, W. (2001) Paramics API Design Document for Actuated Signal, Signal Coordination and Ramp Control. *California PATH Working Paper, UCB-ITS-PWP-2001-11, University of California at Berkeley*.

Liu, H. X., Chu, L, and Recker, W, (2003) "Paramics Plugin Document – Actuated Signal Coordination", PATH ATMS Center, University of California, Irvine, Plugin Compatibility V3/V4, March 20,2003, available on –line at http://www.its.uc.edu/~lchu/documents.

Liu, H., Oh, J. S. and Recker, W. (2002) Adaptive Signal Control System with Online Performance Measure for a Single Intersection. *Transportation Research Record*, 1811**,** 131-138.

Lowrie, P. R. (1982) SCATS Principles, Methodologies, Algorithm, In *IEE Conference on Road Traffic Signal, IEE Publication*, Vol. 207, London, pp. 67-70.

Lucas, D. E., Mirchandani, P. B. and Head, K. L. (2000) Remote Simulation to Evaluate Real-Time Traffic Control Strategies. *Transportation Research Record 1727, pp. 95-100*.

Maher, M.J. (1984) Estimating the Turning Flows at a Junction: A Comparison of Three Models. *Traffic Engineering and Control 25, pp. 19-22*.

McShane, W R, Roess, R. P. and Prassas, E. S. (2004) *Traffic Engineering*, Prentice-Hall, 3rd Edition, Englewood Cliff, N.J.

NTCIP. (2007), 1202-NTCIP Object Definitions for ASC, National Transportation communications for ITS Protocol standards document, order on-line from http://www.ntcip.org.

Nobe, S. (1997) On-line Estimation of Traffic Split Parameters Based on Lane Counts. *Ph.D. Dissertation, the University of Arizona*.

Nelson, E. and Bullock, D. (1998) Quantifying the Impact of Traffic Responsive Signal Systems, http://bridge.ecn.purdue.edu/~darcy/research/trafficresponsive.pdf, Accessed April 1st 2005.

Porche, I. and Lafortune, S. (1997) Adaptive Look-ahead Optimization of Traffic Signals, In *Contorl Group Report* University of Michigan.

Quadstone Limited, (2005) Paramics User Guide Version 5.0, *Quadstone Limited, Edinburgh, U.K.*

Robertson, D. I. and Bretherton, R. D. (1991) Optimizing Networks of Traffic Signals in Real Time – the SCOOT Method. *IEEE Transaction on Vehicular Technology, Vol. 34, pp. 11-15*.

Thomas Riedel (2004a) VS-PLUS Details-Understanding VS-PLUS.

Thomas Riedel (2004b) VS-PLUS Training for Caltrans.

Siemens ITS (2005a) *Eagle EPAC300 Actuated Signal Control*, PIM-177, SIEMENS Energy & Automation, Inc: Business Unit Intelligent Transportation Systems, August 2005

Siemens-ITS, (2005b) *SEPAC Local Controller Software Training Guide Version 1.2*, SIEMENS Energy & Automation, Inc: Business Unit Intelligent Transportation Systems.

Sims, A. G. (1979) The Sydney Coordinated Adaptive Traffic System. *Proceedings of the Engineering Foundation Conference on Research Priorities in Computer Control of Urban Traffic Systems, pp. 12-27*.

Sullia, Ashwin C., (2005) *Development & Evaluation of Traffic Signal Optimization Model with Hardware-in-the-Loop Simulation*, Master's Report, Utah State University, Logan, Utah.

Tarnoff, P. J. (1975) The results of FHWA Urban Traffic Control Research: An Interim Report. *Traffic Engineering, Vol. 45, pp.27-35*.

Venglar, S. and Urbanik, T. (1995) Evolving to Real-Time Adaptive Signal Control, In *Proceedings of the Second World Congress, Intelligent Transportation Systems*, Yokohama, Japan.

Verkehrs-Systeme (2004), VS-NEMA Wizard User Manual, version 1.3.0.

Verkehrs-Systeme (2006),VS-WorkSuite: Product description, available on-line at http://www.VS-PLUS.com/e/vvpP.htm.

Washington, S. P., Karlaftis, M. G., Mannering, F. L. (2003) Statistical and Econometric Methods for Transportation Data Analysis. *Chapman & Hall/CRC*.

Zennaro, Marco, (2008) A framework for the development of traffic control systems. PhD dissertation, University of California at Berkeley, May 2008