PARAMICS Plugin Document – Actuated signal Coordination

Henry X. Liu, Lianyu Chu, Will Recker

PATH ATMS Center
University of California, Irvine

Plugin Compatibility: V3/V4
Release date: 3/20/2003

522 Social Science Tower
Irvine, CA 92697-3600
URL: http://www.its.uci.edu/

Table of Contents

1. Introduction

Coordination is a mode of signal operation designed to allow platoons of traffic to form and "progress" through several signals with minimum stops and delay. Where signals are closely spaced and traffic volumes are high, coordination of signals is necessary to avoid excessive delay and stops.

The actuated signal coordination API inherits most parts of full-actuated signal API, with additional force-off logic to maintain the background cycle length, and form green band for a particular phase (sync phase).

2 Plugin implementation

2.1 Control logic

To provide synchronization and maintain the background cycle length, all coordinated intersection have the same system clock reference point, which is usually the start point of signal coordination plan. For the fixed-time signal coordination plan, there is an offset, which is the difference between two green initiations of the sync phase for two adjacent intersections. However, for the traffic-actuated signal coordination, the sync phase of every coordinated intersection has fixed series of yield points, and the difference between yield points is the background cycle length. These yield points are also local clock reference points to other non-sync phases. The sync phase has minimal bandwidth, i.e. the sync phase has to start at the time of minimal bandwidth earlier than yield point. To do so, all other phases have to be cut at certain points, which are so-called force-off points. These force-off points are usually referenced to the local clock reference point. Figure 4 is the phase diagram of coordinated intersection.

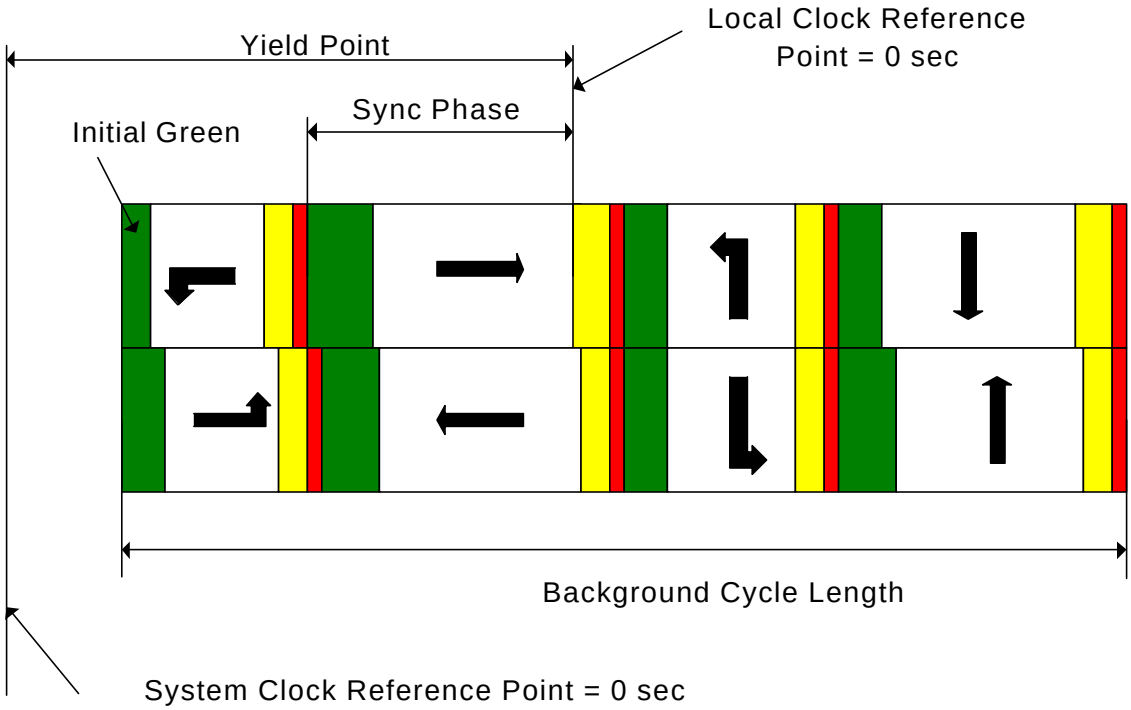


Figure 1. Actuated Signal Coordination

2.2 Control Logic and Pseudo Codes

In order to implement the above concept, the pseudo code for the main control logic is given in the following:

1. Actuated Signal API set up using api_setup( ), includes signal data input, memory allocation, and initial signal phase set up.
2. At every time step, net_action is called:

```
For controller intersection = 1 : n {
  a. Inquiry the current signal information using signal_inquiry( ).
  b. Vehicle presence detection (pp_presence_dection ( ) ).
  c. If (left green time > 0) {
        Check if this phase should be forced off (pp_force_off ( )).
        If (force-off )
            Find the next phase by vehicle presence.
        else {
              excute the current signal plan (pp_excute_plan ( ) )  {
                  If ( left green time < extension &&
                       vehicle presence for extension &&
                        expired green < ( maximal green – extension ) ) {
                            green  time increased by (extension – left green).
                       }
                  If ( left  green time <= time step ) {
                            Find the next phase by vehicle presence.
                   }
              }
         }
     }
     else {
           Amber and red time are counted.
           If ( amber and red time are reached )
                 Set the next signal phase parameters through signal_action( ).
     }
}
```

3. Step-by-step user manual

3.1 Understanding actuated signal coordination

The implemented actuated signal coordination logic has some new concepts. The correct understanding of them is important for the use of the actuated signal coordination plugin. The following is a good description of these terms:

1. Background Cycle Length
To provide synchronization and maintain the background cycle length, all coordinated intersection have the same system clock reference point, which is usually the start point of signal coordination plan.

2. Yield Point

The sync phase of every coordinated intersection has fixed series of yield points, and the difference between yield points is the background cycle length.

3. Sync Phase

These yield points are also local clock reference points to other non-sync phases. The sync phase has minimal bandwidth, i.e. the sync phase has to start at the time of minimal bandwidth earlier than yield point.

4. Force Off

To do so, all other phases have to be cut at certain points, which are so-called force-off points. These force-off points are usually referenced to the local clock reference point.

3.2 Data requirement

As the actuated signal API, two files need to be prepared for the use of signal coordination API. One is the “priorities” file, provided by Paramics, to be used to identify the hierarchy of movements for all phases. The other is the so-called “signal_coordination_control” file, which contains all the signal timing information, intersection layout information, and coordination information.

The following is an example of the part of “signal_coordination_control” file for one intersection.

```
total number of actuated signals is:   4

node 6  ALTON & ICD
```

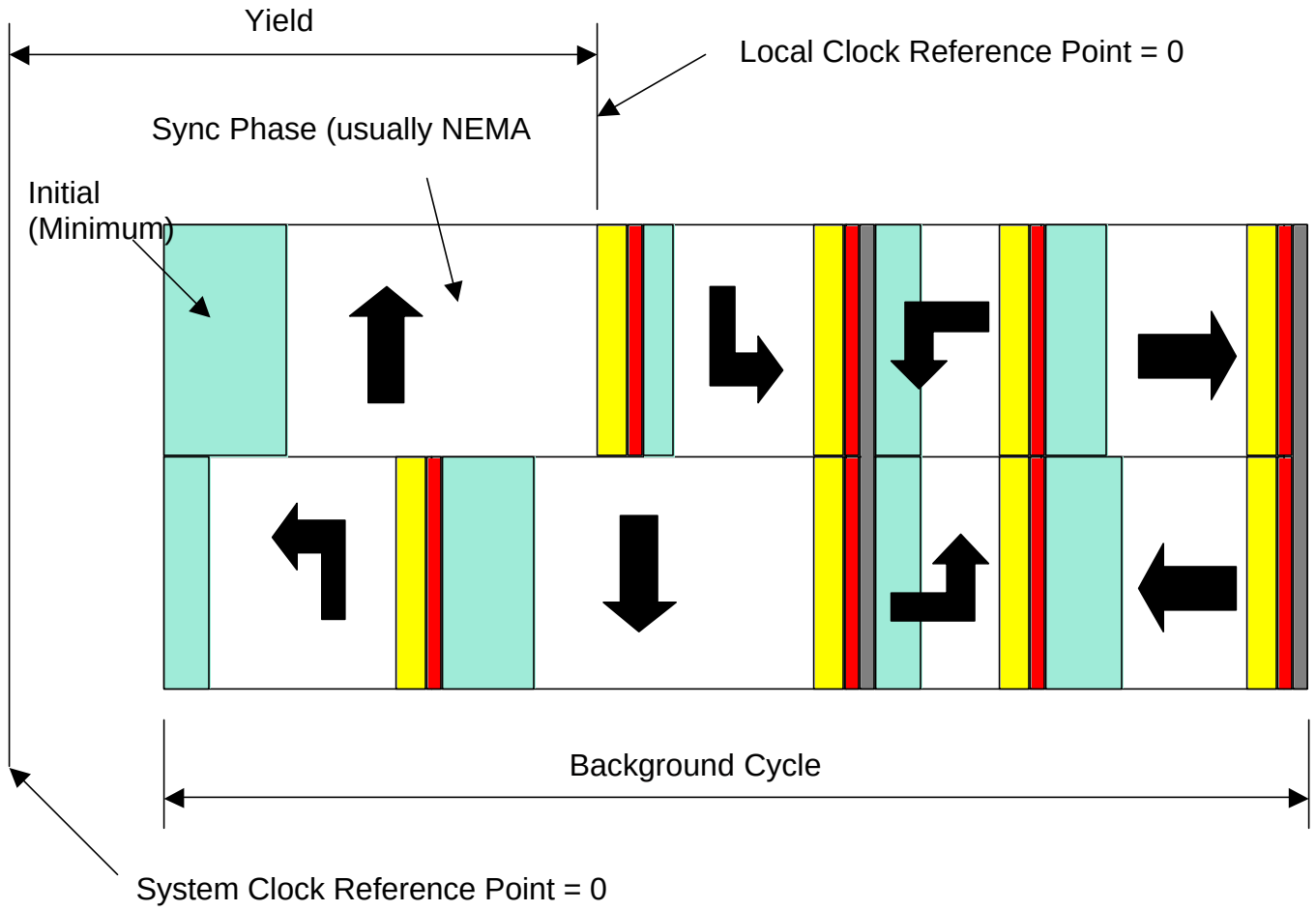
```
movements      1     2     3     4     5     6     7     8
ini_green      5     5     5     5     5     5     5     5
extension      3     5     3     5     3     5     3     5
max_green     24    60    24    32    24    32    24    32
recall         2     6
lanes          2     3     2     3     2     3     2     3
rightturn      1     1     1     1
detector1      aisw   ai2w  ai3w   aiuw
detector2      aiss   ai2s  ai3s   aius
detector3      aise   ai2e  ai3e   aiue
detector4      aisn   ai2n  ai3n   aiun
sync_phase     2     6
cycle_length  60
force_off     36    60    18    27    36    60    18    27
yield_point    5
system_clock   0
```

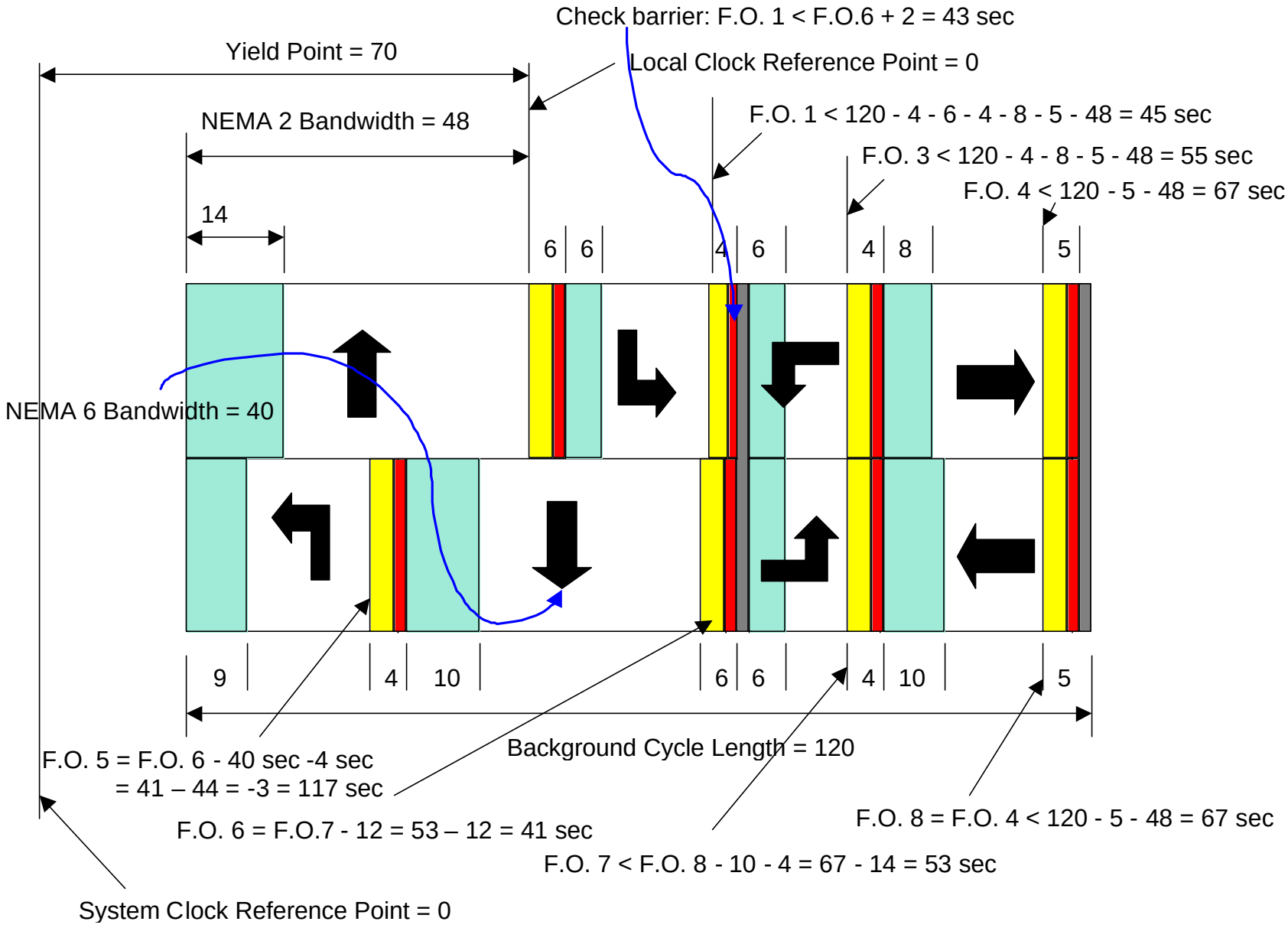
The data for signal coordination has been attached after the intersection layout data for each intersection. Besides to the yield point of the sync phase, all other phases have force-off points, referenced to the local clock reference point. Notice that the maximal green time of primary sync phase has to be the cycle length, since the green time of sync phase may occupy the entire cycle if there is no conflict traffic.

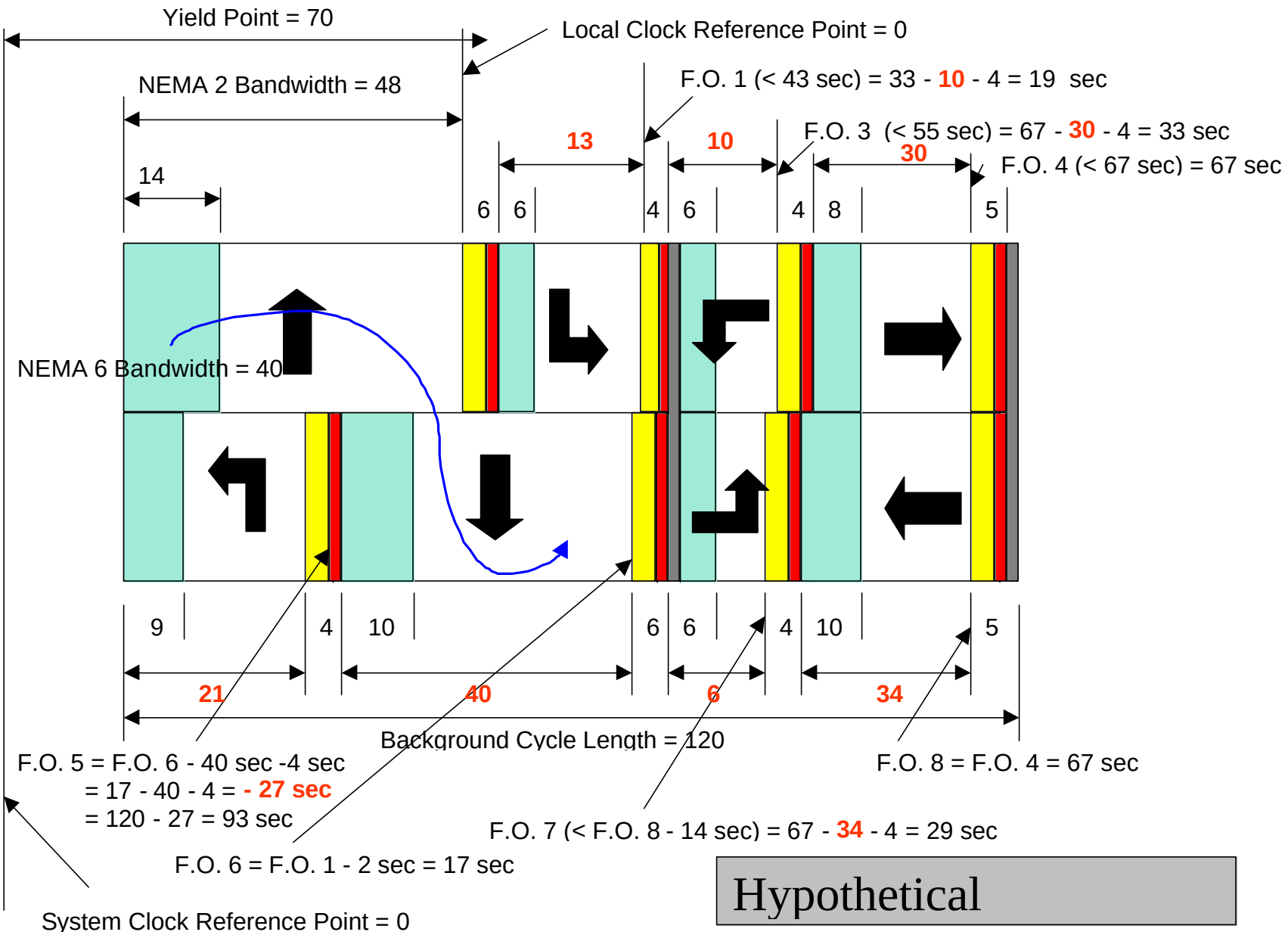
3.3 Examples

Phase Interval Times

Interval	Phase							
	1	2	3	4	5	6	7	8
Walk								
Ped Clear								
Initial	6	14	6	8	9	10	6	10
Extension	2.0	3.0	3.0	2.0	2.0	3.0	3.0	2.0
Max Green	20	50	15	35	20	50	15	35
Yellow	3	5	3	4	3	5	3	4
Red	1	1	1	1	1	1	1	1
Permit	√	√	√	√	√	√	√	√
Max Recall								
Min Recall								
Ped Recall								
Lag Phase	√							


Yield
Local Clock Reference Point = 0
Sync Phase (usually NEMA
Initial
(Minimum)
Background Cycle
System Clock Reference Point = 0


Check barrier: F.O. 1 < F.O.6 + 2 = 43 sec
Yield Point = 70
Local Clock Reference Point = 0
NEMA 2 Bandwidth = 48
F.O. 1 < 120 - 4 - 6 - 4 - 8 - 5 - 48 = 45 sec
F.O. 3 < 120 - 4 - 8 - 5 - 48 = 55 sec
F.O. 4 < 120 - 5 - 48 = 67 sec
14
6
6
4
6
4
8
5
NEMA 6 Bandwidth = 40
9
4
10
6
6
4
10
5
F.O. 5 = F.O. 6 - 40 sec -4 sec
= 41 – 44 = -3 = 117 sec
Background Cycle Length = 120
F.O. 6 = F.O.7 - 12 = 53 – 12 = 41 sec
F.O. 8 = F.O. 4 < 120 - 5 - 48 = 67 sec
F.O. 7 < F.O. 8 - 10 - 4 = 67 - 14 = 53 sec
System Clock Reference Point = 0



Hypothetical

4. Technical Supports

4.1 Contact information

Any comments and suggestions are welcome. Please contact us at the email address: hliu@translab.its.uci.edu.