

# Unity

## Sequence of Events Recorder

### User's Guide

Monaghan Engineering, Inc.  
862 Las Colinas Drive  
Dripping Springs, Texas 78620  
Phone: 1-512-858-4271  
Fax: 1-512-858-1355  
e-mail: [techsupport@monaghan-engineering.com](mailto:techsupport@monaghan-engineering.com)

Publication #10019  
Version 5.00

## TABLE OF CONTENTS

<b>1.0 SER SYSTEM OVERVIEW .....</b>	<b>2</b>
1.1 SER FUNCTION BLOCK.....	2
1.2 GPS FUNCTION BLOCK.....	2
1.3 DELTA TIME FUNCTION BLOCK.....	2
<b>2.0 INSTALLATION.....</b>	<b>2</b>
2.1 MODULE DATABASE .....	3
<b>3.0 SER FUNCTION BLOCK.....</b>	<b>5</b>
3.1 SER FUNCTION BLOCK INPUTS .....	5
3.2 SER FUNCTION BLOCK OUTPUTS .....	8
3.3 EVENT BUFFER STRUCTURE .....	9
3.3.1 EVENT BUFFER TYPE 0.....	9
3.3.2 EVENT BUFFER TYPE 1.....	11
3.3.3 EVENT BUFFER TYPE 2.....	11
<b>4.0 DELTA TIME FUNCTION BLOCK .....</b>	<b>12</b>
4.1 DELTA_T FUNCTION BLOCK INPUTS.....	13
4.2 DELTA_T FUNCTION BLOCK OUTPUTS.....	14
<b>5.0 GPS FUNCTION BLOCK.....</b>	<b>14</b>
5.1 GPS FUNCTION BLOCK INPUTS .....	15
5.2 GPS FUNCTION BLOCK OUTPUTS.....	16

## 1.0 SER SYSTEM OVERVIEW

The Sequence of Events Recorder (SER) provides a permanent record of events that occur within milliseconds of each other, such as the operation of circuit breakers or the shutdown sequences of compressors and other high speed devices. The SER will provide the time of the event, the state of the point, and the point identification.

Multiple SER cards may be synchronized together to provide a distributed event recording system. Interface cards are available to synchronize the SER cards with an external time standard such as a Global Positioning Satellite (GPS) receiver.

### 1.1 SER FUNCTION BLOCK

The SERx53 function block provides an easy means of both configuring and gathering data from the SER cards. When the controller is reset, the function block will configure the SER cards and scan for event data. When an event is detected, the function block will build a buffer containing the event data and either send it over Modbus Plus to a receiving drop, or hold the buffer until a host computer polls the PLC for the data.

### 1.2 GPS FUNCTION BLOCK

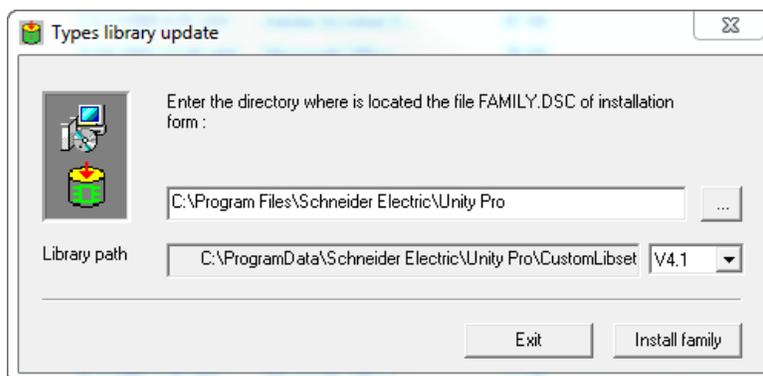
The GPS function block will continuously update a set of time registers from the GPS or IRIG-B time source. When connected to a GPS receiver it will also download all of the GPS location and velocity data. The function block will also synchronize the PLC time of day clack with the GPS or IRIG-B time source.

### 1.3 DELTA TIME FUNCTION BLOCK

The DELTA T Function Block is used to calculate the time difference between two events and set an alarm if the time exceeds a specified value. The block also maintains a history of the past 10 delta time measurements.

## 2.0 INSTALLATION

The MEI Event Recording library must be added to the Unity database. This is done by running the Types Library Update.



Use the browse button to go to the cn\_MEI folder and select family.dsc. Then press Install Family. This will install the library in the Unity database.

## 2.1 MODULE DATABASE

If you have not previously entered the SER and GPS modules into the Unity Hardware Catalog you can do so at this time.

The SER x53 00 and GPS 100 00 cards are supported under Unity version 2.1 and later. In Unity versions 4.0 and earlier the GPS 100 00 card is entered in the project configuration as a Generic Analog card and the SERx53 00 card is entered in the project configuration as a Generic Digital card. The following parameters should be entered into the card configuration screen.

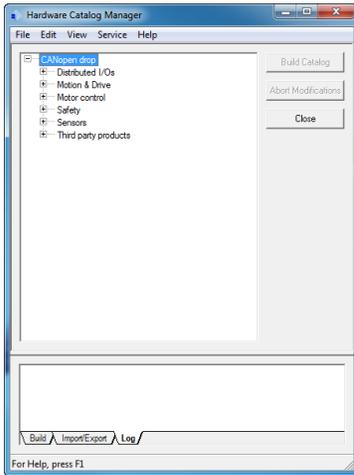
### GPS 100 00

Module Personality	307
Mapping	Word(%I-3X %M-4X)
Setting the Module	
Number of Inputs	12
984 Quantum Input Format (Simple Module)	0
984 Quantum Input Format (DPM Module)	0
Number of Outputs	6
984 Quantum Output Format (Simple Module)	0
984 Quantum Output Format (DPM Module)	0
Input Type	Binary
Input Starting Address	Enter the starting 4X address of the card. Example: 4X0001 = 1
Output Type	Binary
Output Starting Address	Enter the starting 3X address of the card. Example: 3X0001 = 1

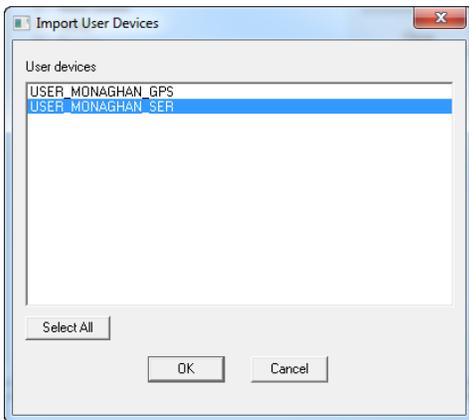
### SER x53 00

Module Personality	302
Mapping	Bit(%I-1X %M-0X)
Setting the Module	
Number of Inputs	8
984 Quantum Input Format (Simple Module)	0
984 Quantum Input Format (DPM Module)	0
Number of Outputs	8
984 Quantum Output Format (Simple Module)	0
984 Quantum Output Format (DPM Module)	0
Input Type	Binary
Input Starting Address	Enter the starting 1X address of the card. Example: 1X0001 = 1
Output Type	Binary
Output Starting Address	Enter the starting 0X address of the card. Example: 0X0001 = 1

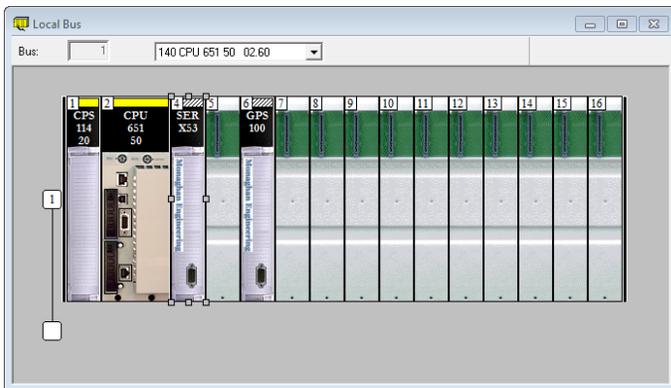
Starting with Unity 4.1, the GPS 100 00 and SERx53 00 card can be added to the Hardware Catalog. This is done by running the Hardware Catalog Manager.



Select File and then Import User Devices. Open the USER\_MONAGHAN.cpx file.



Select All and press OK. This will add the GPS 100 00 and SERx53 00 cards to the Unity Hardware Catalog. When configuring the drop, the GPS 100 00 and SERx53 00 cards will be shown in the rack.



Double clicking the cards will bring up the configuration screen where you can enter the proper starting addresses for each card.

### 3.0 SER FUNCTION BLOCK

The SER Function block will support up to 16 SER cards. Additional blocks may be added if there are more than 16 cards.

For every SER module mounted in the PLC, a variable of the type SerCardDat is required. This variable is used to define all of the configuration parameters for the card. The SER block will use this information to download configuration data to the card and to retrieve event data from the card.

The SER block is also responsible for managing a common buffer into which events are placed. When an event buffer is ready to be read, the SER function block will turn on the BUF\_RDY output indicating that data is ready. This output can either be polled by master computer and the data gathered when the SER block indicates a buffer is ready, or the output may be connected to a MSTR block which will automatically send the event data over Modbus Plus to a BM85E Bridge Mux, or a PC with a SA85 card. After the data has been collected, the BufAck input is used to tell the SER block to clear the buffer and to start collecting new event data.

The SER function block will also set the PLC time of day clock from the time in the SER card, or set the time in the SER card from the PLC time of day clock.

		SER_X53			
INT	---	PLC_NUM			
INT	---	DELAY			
INT	---	BUF_TYPE	BUF_RDY	---	BOOL
BOOL	---	BUF_ACK	EVENT_CT	---	INT
BOOL	---	SYNC_MTR	BUFFER	---	SER_BUFFER_DEF
BOOL	---	CONFIG	ERROR	---	BOOL
BOOL	---	SET_PLC	L_RDY	---	BOOL
BOOL	---	SET_SER	L_BUF	---	LOCAL_BUF_DEF
SER_CARD_DAT	---	CARD_1			
	:				
SER_CARD_DEF	---	CARD_16			

#### 3.1 SER FUNCTION BLOCK INPUTS

##### PLC\_NUM

The value of the PLC\_NUM input is placed in the output buffer to indicate which PLC the buffer is from.

##### DELAY

The value of the Delay input is the number of 10mS intervals the block will wait from the last event until the BufRdy output is set true.

##### BUF\_TYPE

The value of BUF\_TYPE is used to determine the structure of the output buffer. There are currently three different buffer types defined. The format of each buffer type is described in section 3.3 of this document.

##### BUF\_ACK

A true value on the BUF\_ACK input will clear the event buffer and set the BUF\_RDY output false.

**SYNC\_MTR**

A true value on the SYNC\_MTR input will configure the first SER card as a time sync master.

**CONFIG**

A rising edge on the CONFIG input will download the configuration data to all of the cards.

**SET\_PLC**

A rising edge on the SET\_PLC input will set the PLC Time of Day clock from the first SER card.

**SET\_SER**

A rising edge on the SET\_SER input will set the first SER cards clock from the PLC Time of Day clock.

**CARD\_1 – CARD\_16**

The block will accept up to 16 card inputs, which expect a data type of the SerCardDat type.

## TYPE

```
SerCardDat: STRUCT
    Address0X:          INT;
    Address1X:          INT;
    Enable:             BOOL;
    Config1:            BOOL;
    ConfigPor:          BOOL;
    FilterInput:        BOOL;
    BatteryEnable:      BOOL;
    Point:              SerPoint;
    TimeQuality:        INT;
    EventCount:         INT;
    FirmwareVersion:    INT;
    CardError:          INT;
    SyncLock:           BOOL;
END_STRUCT;
```

**Address0X - Address of First 0x Input in I/O Map**

The SERX53 function block must write the commands to the SER module through 0x registers. . The number entered here is an offset from 0X0000. Register 0X00001 is addressed as 1. Please note that the address entered here must match a 0x address as entered in the PLC's I/O map for the SER module.

**Address1X - Address of First 1x Input in I/O Map**

The SERX53 function block must read the values from the SER module through 1x inputs. The number entered here is an offset from 1X0000. Register 1X0001 is addressed as 1. Please note that the address entered here must match a 1x address as entered in the PLC's I/O map for the SER module.

**Enable – SER Card Enable**

A true value will enable event recording on the SER card. Any unused card inputs on the function block should have a card variable with this value set False.

**Config1 – Configure All Points The Same As Point 1**

For a FALSE value, all 32 points are configured with the parameters entered individually for every point. However, if all 32 points operate with the same parameters, these parameters need to be entered only once for point 1. For a TRUE value, the parameters for point 1 are used to configure all 32 points.

**ConfigPor – Configure Card After Power On Reset**

Since the SER module cannot preserve any parameterizations through a power cycle, all point

parameters are reset to their default values after power is cycled. If this value is TRUE, the function block will reconfigure the SER card after a Power On Reset.

#### **FilterInput – Use Input Filter On 1X Registers**

For a FALSE value, the 1X register input points will be passed to the processor immediately when they change. For a TRUE value, the filter value for each point will be used to filter the 1X register inputs before they are passed to the processor.

#### **BatteryEnable – Enable the Battery Backed Up Memory**

For a FALSE value, the card firmware will not save event data when the card is powered down. For a TRUE value, the card firmware will retain old event data and generate a restart event when the card receives power. This is a future option that is not supported by the SER hardware at this time.

#### **Point – SER point Data**

An array of variables of the PointData type which define the configuration of each input point.

```

TYPE
    PointData: STRUCT
        Filter:          UINT;
        Debounce:       UINT;
        Chatter:        BYTE;
        OffScan:        BOOL;
    END_STRUCT;

```

#### **Filter – SER Card Input Filter**

The purpose of the contact filter time is to eliminate false event messages caused by noise. The filter time is the amount of time that a point must stay in a new state in order to be recognized as an event. As an example, if the filter time for a point is set to 25 milliseconds, then when the input changes state it must remain in the new state for 25 milliseconds before an event message is generated. If the input returns to the previous state in less than 25 milliseconds, the filter is reset and the next time the state changes it will again have to remain in that state for 25 milliseconds before an event message is generated. The time that is associated with the event is the time at which the point first changed, not the time when the filter recognized the event. If two input points that have different input filter values were to change at the same time, the event messages would be generated at different times but the time that was contained in the event message would be identical.

#### **Debounce – SER Card Input Debounce**

The purpose of the contact debounce time is to prevent multiple events from being generated from a single contact closure. The debounce time is the amount of time that input processing is disabled for a point after an event has been recorded.

#### **Chatter – SER Card Input Chatter**

This parameter determines the maximum number of status changes allowed per minute before the point is taken off scan. Once a point is taken off scan, it will be put on scan again after being at least one minute below the chatter count.

#### **OffScan – SER Point Scan State**

If the offscan value is TRUE the SER card will not process SER data for that point.

#### **TimeQuality – Current Time Quality – READ ONLY**

This register contains the current quality of the SER time stamps. The values are as follows:

Value	Quality
0	Good, accuracy is GMT +/- 1ms
1	Fair, accuracy is GMT +/- 50ms
2	Poor, accuracy is GMT +/- 50ms or worse
3	Bad, no time reference available

**EventCount – SER Card Event Count – READ ONLY**

The number of events currently in the SER card buffer.

**FirmwareVersion – Firmware Version of the SER Card – READ ONLY**

This register contains the version of the firmware in the SER card.

**CardError – Error Status of Card – READ ONLY**

This register contains the SER Card error. Its primary purpose is to aid debugging during the programming and testing of the program. Whenever a bit is turned on, a certain error condition is present.

Bit	Error Condition
15	The 0x address entered in the function block is outside the valid 0x addresses as configured in the PLC's configuration. For example, if the inputs 10001 through 10512 are configured, the highest valid 1x address would be 10449, since the SER module requires 64 1x inputs.
14	The 0x address entered in the function block is outside the valid 0x addresses as configured in the PLC's configuration. For example, if the outputs 00001 through 01536 are configured, the highest valid 0x address would be 01473, since the SER module requires 64 0x outputs.
8	The Time of Day Clock must be configured.

**SyncLock – Time Signal Synchronization – READ ONLY**

This register contains the current sync status for the SER card. This bit follows the indication of the Red F light on the front of the SER module. For a TRUE value, the card is receiving time sync messages and the Red F light is off. For a FALSE value, the card is not receiving time sync messages and the Red F light is on.

**3.2 SER FUNCTION BLOCK OUTPUTS****BUF\_RDY**

The BUF\_RDY output is set true when an event buffer is ready to be read.

**EVENT\_CT**

The EVENT\_CT output contains the number of events in the event buffer.

**BUFFER**

The BUFFER output expects a data type of the SerBuffer type which is defined as follows:

TYPE

```
SerBuffer: STRUCT
    PlcNumber:          INT;
    BufferType:         INT;
    NumberOfEvents:    INT;
    Nul1:              INT;
    Nul2:              INT;
    Nul3:              INT;
    Nul4:              INT;
```

```

        Nul5:          INT;
        Nul6:          INT;
        Version:      INT;
        Events:       ARRAY [0..89] OF UINT;
    END_STRUCT;

```

**ERROR**

The ERROR output is set true when there is a configuration error. Refer to the CardError variable in each card definition structure for detailed error information.

**L\_RDY**

The L\_RDY output is set true for one scan cycle time when a new event is placed in the Local Buffer.

**L\_BUF**

The L\_BUF output expects a buffer of the LocalBuf type, which is defined as follows:

```

TYPE
    LocalBuf: STRUCT
        EventType:    INT;
        Card:         INT;
        Point:        INT;
        DateTime:     UDINT;
        Milliseconds: INT;
        Quality:      INT;
        CurrentState: BOOL;
    END_STRUCT;

```

The local buffer provides a second port for retrieving event data. The event data is only guaranteed to be valid for one scan cycle time. This port is used by other function blocks that need access to event data.

**3.3 EVENT BUFFER STRUCTURE**

The event buffer consists of two parts, the header and the buffer proper. The header is located in the first 10 registers. The first register contains the PLC identification, the second register contains the type of event buffer and the third register contains the number of events following. The tenth register contains the SER\_x53 function block version number. The other six registers are presently unused.

**3.3.1 EVENT BUFFER TYPE 0**

The events are stored in an array of registers immediately following the header. The first event will be stored in registers 11 through 13, the second event in registers 14-16, and the nth event in registers  $(n-1)*3+10$  through  $(n-1)*3+12$ . Up to 30 events can be stored in a single buffer. The three registers per event are used as follows:

Register	Bits	Usage
1	11-15	SER Card Identification (0..31) (from SER Card Function Block)
1	10	Current Status Value (0 or 1, from SER module)
1	5-9	Status Identifier (0..31, from SER module)
1	0-4	Event Type <ul style="list-style-type: none"> <li>1: Status Change</li> <li>2: Point On Scan</li> <li>3: Point Off Scan</li> <li>4: Chatter On Scan</li> <li>5: Chatter Off Scan</li> <li>6: Power On Reset</li> <li>7: External Time Sync Lock</li> <li>8: External Time Sync Lost</li> <li>9: SER Card Event Buffer Overflow (from SER module)</li> <li>10: Scan Buffer Overflow</li> <li>11: Time Resync Old Time</li> <li>12: Time Resync New Time</li> <li>13: Hourly Time Update</li> <li>14: Time Resync New Date</li> <li>15: Reconfigure</li> <li>16: Output Point Change</li> <li>17: Restart Date</li> <li>18: Restart Time</li> </ul>
		Events 1 – 12, 16, 18
2	10-15	Seconds (0..59, from SER module)
2	0-9	Milliseconds (0..999, from SER module)
		Events 13 – 15, 17
2	9-13	Hour (0..23)
2	4-8	Day (1..31)
2	0-3	Month (1..12)
		Events 1 – 12, 16, 18
	14-15	Time Quality
3	8-12	Hours (0..23, from SER module)
3	0-5	Minutes (0..59, from SER module)
		Events 13 – 15, 17
3	14-15	Time Quality
3	0-12	Year (0..4095)

Example:

Offset	Value	Description
0	23	Message originated in PLC 23
1	x	unused
2	2	Message contains two events
3	x	unused
4	x	unused
5	x	unused
6	x	unused
7	x	unused
8	x	unused
9	100	Loadable version 1.00
10	15873 = 00111 1 10000 00001	Identifier of first event SER card 7 Current status 1 Status point 16 Event type 1 (status change)
11	39228 = 100110 0100111100	Second/milliseconds of event 38 seconds/316 milliseconds
12	4399 = 000 10001 00 101111	Hours/minutes of event 17 hours/47 minutes => Status point 16 on SER card 7 of PLC 23 changed to value 1 at 17:47:38.316
13	15873 = 00101 0 00000 01001	Identifier of second event SER card 5 Current status not used Status point not used Event type 9 (SER buffer overflow)
14	39228 = 100110 0101110010	Second/milliseconds of event 38 seconds/370 milliseconds
15	4399 = 000 10001 00 101111	Hours/minutes of event 17 hours/47 minutes => Buffer overflow on SER card 5 of PLC 23 at 17:47:38.370

**3.3.2 EVENT BUFFER TYPE 1**

The events are stored in an array of registers immediately following the header. The buffer will only contain a single event and each data value will be placed in a separate register.

Register	Usage
1	Event Type
2	Point
3	Current State
4	Card
5	Millisecond
6	Second
7	Minute
8	Hour
9	Day
10	Month
11	Year
12	Quality

**3.3.3 EVENT BUFFER TYPE 2**

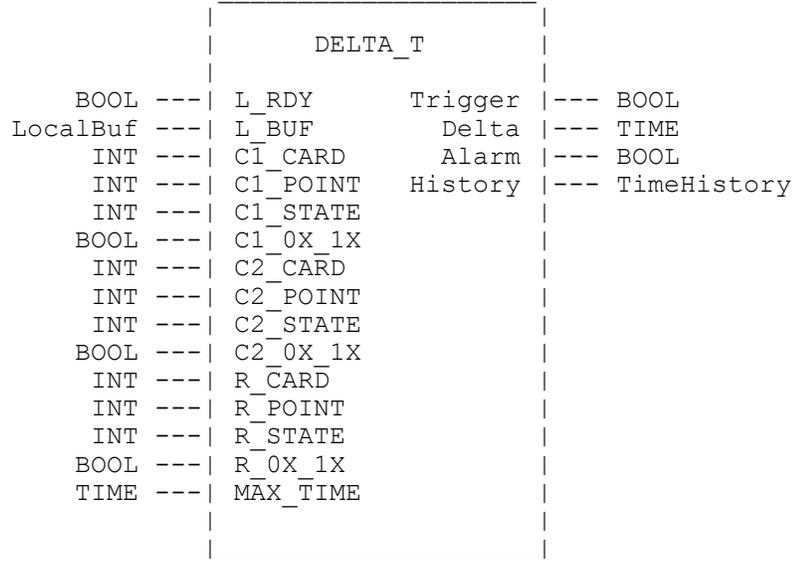
The events are stored in an array of registers immediately following the header. Each event takes four registers. The first event will be stored in registers 11 through 14, the second event in registers 15-18, and the nth event in registers (n-1)\*4+10 through (n-1)\*4+13. Up to 22 events can be stored in a single buffer. The four registers per event are used as follows:

Register	Bits	Usage
1	11-15	SER Card Identification (0..31) (from SER Card Function Block)
1	10	Current Status Value (0 or 1, from SER module)
1	5-9	Status Identifier (0..31, from SER module)
1	0-4	Event Type <ul style="list-style-type: none"> <li>1: Status Change</li> <li>2: Point On Scan</li> <li>3: Point Off Scan</li> <li>4: Chatter On Scan</li> <li>5: Chatter Off Scan</li> <li>6: Power On Reset</li> <li>7: External Time Sync Lock</li> <li>8: External Time Sync Lost</li> <li>9: SER Card Event Buffer Overflow (from SER module)</li> <li>10: Scan Buffer Overflow</li> <li>11: Time Resync Old Time</li> <li>12: Time Resync New Time</li> <li>13: Hourly Time Update</li> <li>14: Time Resync New Date</li> <li>15: Reconfigure</li> <li>16: Output Point Change</li> <li>17: Restart Date</li> <li>18: Restart Time</li> </ul>
2	0-9	Milliseconds (0..999, from SER module)
2	14-15	Time Quality
3	0-15	Date Time LSB (32 Bit Long Integer. Number of seconds since 1/1/1984)
4	0-15	Date Time MSB (32 Bit Long Integer. Number of seconds since 1/1/1984)

#### 4.0 DELTA TIME FUNCTION BLOCK

The DELTA T Function Block is used to calculate the time difference between two events and set an alarm if the time exceeds a specified value.

The block will monitor the event stream from the local buffer of the SERX53 block. When the DELTA\_T block detects an event that matches the C1\_CARD, C1\_POINT, C1\_STATE and C1\_0X\_1X inputs, or the C2\_CARD, C2\_POINT, C2\_STATE and C2\_0X1X inputs, the trigger output will turn on, the Delta output will be set to zero and an internal timer will be started. Two command inputs are provided so that delta time measurements can be made on a piece of equipment that has inputs for both local and remote operation. When the block detects an event that matches the R\_CARD, R\_POINT, R\_STATE and R\_0X1X inputs, the trigger output will be turned off, the time difference between the command and response events will be calculated and placed in the Delta output and if the time exceeds the MAX\_TIME input, the Alarm output will be turned on. If the block does not receive a response event, when the internal timer exceeds the MAX\_TIME input, the Alarm output will be turned on and the Trigger output will remain on.



#### 4.1 DELTA\_T FUNCTION BLOCK INPUTS

##### L\_RDYy

The L\_RDY input is set true for one scan cycle time when a new event is available in the Local Buffer.

##### L\_BUF

The L\_BUF input expects a buffer of the LocalBuf type, which is defined as follows:

```

TYPE
  LocalBuf: STRUCT
    EventType:      INT;
    Card:           INT;
    Point:          INT;
    DateTime:       UDINT;
    Milliseconds:   INT;
    Quality:        INT;
    CurrentState:   BOOL;
  END_STRUCT;
    
```

##### C1\_CARD

This is the card number that the block will look for to trigger the time measurement. (1 - 16)

##### C1\_POINT

This is the point number that the block will look for to trigger the time measurement. (1- 32)

##### C1\_STATE

This is the state that the block will look for to trigger the time measurement.

##### C1\_0X\_1X

A zero on this input will force the block to look for an event type 16, Output Change Event. A one on this input will force the block to look for an event type 1, Input Change Event.

##### C2\_CARD

This is the card number that the block will look for to trigger the time measurement. (1 - 16)

**C2\_POINT**

This is the point number that the block will look for to trigger the time measurement. (1- 32)

**C2\_STATE**

This is the state that the block will look for to trigger the time measurement.

**C2\_0X1X**

A zero on this input will force the block to look for an event type 16, Output Change Event. A one on this input will force the block to look for an event type 1, Input Change Event.

**R\_CARD**

This is the card number that the block will look for to complete the time measurement. (1 - 16)

**R\_POINT**

This is the point number that the block will look for to complete the time measurement. (1 – 32)

**R\_STATE**

This is the state that the block will look for to complete the time measurement.

**R\_0X1X**

A zero on this input will force the block to look for an event type 16, Output Change Event. A one on this input will force the block to look for an event type 1, Input Change Event.

**MAX\_TIME**

The maximum time allowed between events before the alarm output is turned on.

**4.2 DELTA\_T FUNCTION BLOCK OUTPUTS****TRIGGER**

The TRIGGER output is set true when the command event is detected.

**DELTA**

The time difference between the command event and the response event.

**ALARM**

The ALARM output is set true if the time difference between the command event and the response event exceeds the MAX\_TIME input.

**HISTORY**

A structure of the DeltaHistory type that contains the last 16 Delta Time measurements and an average of the last 16 measurements.

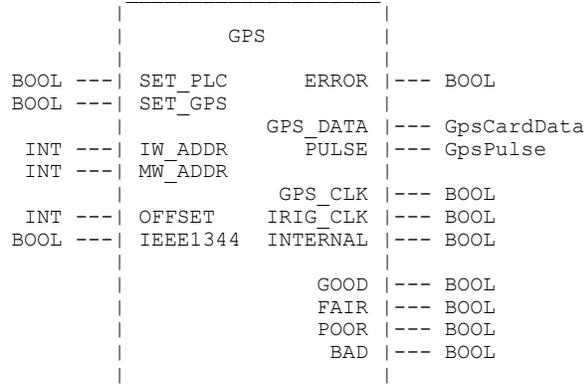
TYPE

```
DeltaHistory: STRUCT
    History:          ARRAY [0..15] OF TIME;
    Average:          TIME;
END_STRUCT;
```

**5.0 GPS FUNCTION BLOCK**

The GPS function block performs 3 different functions:

1. When SET\_PLC input to the block is active the function block will transfer the GPS 100 00 time to the PLC time of day clock.
2. When SET\_GPS input to the block is active or when the GPS function block detects that the GPS 100 00 card has gone through a power on reset sequence the function block will transfer the PLC time of day clock to the GPS 100 00 card.
3. The GPS function block continuously updates the time registers, GPS receiver status registers and the GPS position and velocity registers.



### 5.1 GPS FUNCTION BLOCK INPUTS

#### SET\_PLC

The SET\_PLC input is a Boolean input. A rising edge on the SET\_PLC input will cause the PLC time of day clock to be set from the GPS 100 card.

#### SET\_GPS

The SET\_GPS input is a Boolean input. A rising edge on the SET\_GPS input will cause the GPS 100 card time to be set from the PLC time of day clock.

#### IW\_ADDRESS- Address of First 3x Input in I/O Map

The GPS function block must read the values from the GPS module through 3x inputs. The number entered here is an offset from 3X0000. Register 3X0001 is addressed as 1. Please note that the address entered here must match a 3x address as entered in the PLC's I/O map for the GPS module.

#### MW\_ADDRESS - Address of First 4x Input in I/O Map

The GPS function block must write the commands to the GPS module through 4x registers. . The number entered here is an offset from 4X0000. Register 4X0001 is addressed as 1. Please note that the address entered here must match a 4x address as entered in the PLC's I/O map for the GPS module.

#### OFFSET

The offset to be added to the time received from the GPS or IRIG-B inputs. This allows the time to be converted to local time. The value can range from + 23 hours to – 23 hours. This value is written to the first 4X register addressed by the card.

#### IEEE1344

The standard IRIG-B time code signal does not include year information. As a result, the function block will use the year from the PLC time of day clock to update its clock when it is locked on to an IRIG-B

source. The IEEE1344 extension to the IRIG-B specification encodes the year in the IRIG-B signal. When this input is set True, the function block will decode the year information from the IRIG-B signal. This option is not currently supported by the GPS 100 00 hardware.

## 5.2 GPS FUNCTION BLOCK OUTPUTS

### ERROR

The ERROR output is set true when there is a configuration error. Refer to the GPS\_DATA structure for detailed error information.

### GPS\_DATA

The GPS\_DATA output expects a data type of GpsCardData which is defined as follows:

```
typedef struct GpsCardData
{
    int    EfbError;
    int    DayOfWeek;
    int    Month;
    int    Day;
    int    Year;
    int    Hour;
    int    Minute;
    int    Second;
    int    Millisecond;
    int    LongYear;
    int    EfbVersion;
    int    FirmwareVersion;
    int    GpsStatus;
    int    NumSats;
    float  Latitude;
    float  Longitude;
    float  Altitude;
    float  EastVelocity;
    float  NorthVelocity;
    float  UpVelocity;
    int    TimeQuality;
    bool   TimeGood;
    bool   TimeFair;
    bool   TimePoor;
    bool   TimeBad;
    bool   GpsClock;
    bool   IrigClock;
    bool   InternalClock;
};
```

#### EfbError - GPS Function Block Error Status

This register contains the GPS function block error status. Its primary purpose is to aid debugging during the programming and testing of the ladder logic program. Whenever a bit is turned on, a certain error condition is present.

Bit	Error Condition
15	The Time of Day Clock has not been configured in the PLC's configuration.
13	An invalid 4X register address has been entered.
12	An invalid 3X register address has been entered.

#### DayOfWeek - Day of Week

1 = Sunday, 2 = Monday, ... 7 = Saturday

#### Month - Month of Year

The current month of the year.

**Day - Day of Month**

The day of the month

**Year – Year**

The current year. (0 – 99)

**Hour – Hour**

The current hour.

**Minute – Minute**

The current minute.

**Second – Second**

The current second.

**Millisecond – Millisecond**

The current millisecond.

**LongYear – Year**

The current year as a 16 bit integer. ( 1984 – 65536)

**EfbVersion – EFB Function Block Version Number**

This register contains the GPS function block version number.

**FirmwareVersion - GPS 100 00 Card Firmware Version Number**

This register contains the GPS 100 00 card firmware version number.

**GpsStatus - GPS Status**

The GPS status register contains information about the satellite tracking status of the GPS receiver.

0 = Fix not available

1 = Non-differential GPS fix available

2 = Differential GPS fix available

**NumSats – Number of Satellites**

The number of satellites currently being tracked by the GPS receiver.

**Latitude - Latitude Position (Single Precision)**

The current latitude in +/- ddmm.mmmm format. Positive being North latitude and negative being South latitude.

**Longitude – Longitude Position (Single Precision)**

The current longitude in +/- dddmm.mmmm format. Positive being East longitude and negative being West longitude.

**Altitude - Altitude Position (Single Precision)**

The antenna height above or below mean sea level, -9999.9 to 99999.9 meters.

**EastVelocity - East Velocity (Single Precision)**

Not supported by the GPS 300 00 receiver.

**NorthVelocity - North Velocity (Single Precision)**

Not Supported by the GPS 300 00 receiver.

**UpVelocity - Up Velocity (Single Precision)**

Not Supported by the GPS 300 00 receiver.

**TimeQuality - Time Quality**

The current time quality.

0 = Good

1 = Fair

2 = Poor

3 = Bad.

**TimeGood**

The TimeGood output is of the type Boolean. It will be true when the time quality is good.

**TimeFair**

The TimeFair output is of the type Boolean. It will be true when the time quality is fair.

**TimePoor**

The TimePoor output is of the type Boolean. It will be true when the time quality is poor.

**TimeBad**

The TimeBad output is of the type Boolean. It will be true when the time quality is bad.

**GpsClock**

The GpsClock output is of the type Boolean. It will be true when the GPS 100 card is using the GPS receiver as the time source.

**IrigClock**

The IrigClock output is of the type Boolean. It will be true when the GPS 100 card is using the IRIG-B time code signal as the time source.

**InternalClock**

The InternalClock output is of the type Boolean. It will be true when the GPS 100 card is using its crystal oscillator as the time source.

**PULSE**

The PULSE output expects a data type of GpsPulse which is defined as follows:

```
typedef struct GpsPulse
{
    bool    Second;
    bool    Minute;
    bool    Hour;
    bool    Day;
    bool    Week;
    bool    Month;
    bool    Year;
};
```

**Second**

This One Pulse per Second output will transition from a 0 to a 1 for one scan cycle at the start of every second.

**Minute**

This One Pulse per Minute output will transition from a 0 to a 1 for one scan cycle at the start of every minute.

**Hour**

This One Pulse per Hour output will transition from a 0 to a 1 for one scan cycle at the start of every hour.

**Day**

This One Pulse per Day output will transition from a 0 to a 1 for one scan cycle at the start of every day.

**Week**

This One Pulse per Week output will transition from a 0 to a 1 for one scan cycle at the start of every week.

**Month**

This One Pulse per Month output will transition from a 0 to a 1 for one scan cycle at the start of every month.

**Year**

This One Pulse per Year output will transition from a 0 to a 1 for one scan cycle at the start of every year.