Chipkin Automation Systems

Driver Manual

(Supplement to the FieldServer Instruction Manual)

FS-8705-21

Integlex FACP Serial Driver

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after Aug 20, 2012

, 3381 Cambie St- #211, Vancouver, BC, Canada, V5Z 4R3

Tel: (866) 383-1657, **Fax:** (416) 915-4024

Email: dfs@chipkin.com **Website:** www.chipkin.com

TABLE OF CONTENTS

TABLE OF CONTENTS **2**

1 Integlex FACP Driver Description **5**

2 Driver Scope of Supply **6**

2.1 Supplied with this driver 6

2.2 Provided by the Supplier of 3rd Party Equipment 6

2.2.1. Required 3rd Party Hardware *6*

3 Hardware Connections **7**

3.1 Hardware Connections 7

3.2 Block Diagram 8

3.3 Recommended Cable Assembly 9

4 Configuring the FieldServer as a Integlex Passive Client **10**

4.1 Data Arrays 11

4.1.1 Data Arrays - Example *11*

4.2 Client Side Connections 12

4.2.1 Client Side Connection Descriptions - Example *13*

4.3 Client Side Nodes 14

4.3.1 Client Side Nodes - Example *14*

4.4 Client Side Map Descriptors 15

4.4.1 FieldServer Related Map Descriptor Parameters *15*

4.4.2 Driver Related Map Descriptor Parameters *16*

4.5 How Data is Stored / Summarizing Events 20

4.5.1 Classification 20

4.5.2 System/Panel Events 20

4.5.3 Device / Module Events 24

4.5.4 Summarizing Data 24

4.5.5 MultiState Data 27

4.5.6 On Reset 28

4.6 Some More Examples 29

4.6.1 Map Descriptor Example 1 – Store System Troubles 29

4.6.2 Map Descriptor Example 2 – Store Panel Troubles 30

4.6.3 Map Descriptor Example 3 – Store Classifications 31

4.6.4 Map Descriptor Example 4 – Store Device Event Data 32

4.6.5 Map Descriptor Example 5 – Store Device Event Data 33

4.6.6 Map Descriptor Example 6 – Summarizing Data 34

4.6.7 Map Descriptor Example 7 – Summarizing Data (Full) 35

4.6.8 Map Descriptor Example 8 – MultiState Data 37

4.6.9 Map Descriptor Example 9 – Simplified Config for MultiState Data 39

5 Configuring the FieldServer as a Integlex FACP Server 41

Appendix 1.1. System Reset / Synch Panel and Gateway 42

Appendix 1.2. Loading New Classifications 43

Appendix 1.3. Loading New System/Panel Troubles Event Types 44

Appendix 1.4. Unsupported Features 47

Appendix 1.5. Zone Data 47

Appendix 1.6. Simulation Data 47

Appendix 1.7. Driver Error Messages 48

Appendix 1.8. Exposing Driver Stats 55

Appendix 1.9. Revision History 58

1 Integlex FACP Driver Description

The Integlex FACP protocol driver can be used to connect to suitably enabled Integlex FACP panels from Nohmi Bosai Ltd.

The Integlex driver is a passive client driver intended for connection to the serial port of an Integlex Fire Alarm Control Panel (FACP). A passive client driver waits for messages to be sent to it (by the panel). The driver cannot send messages to the panel and hence it cannot request the state of any point in the panel.

The driver can process alarm and trouble events, system troubles and some other messages from the panel.

The driver can only be used as a client. Minimal server functionality is provided only to support our ongoing quality assurance program by facilitating automated testing of the driver. Server operation is not documented or supported.

Synchronization

A consequence of the fact that this is a passive client driver, is that the FieldServer must be synchronized with the panel by restarting the FieldServer and then resetting the panel to reset the data.

Max Nodes Supported

FieldServer Mode	Nodes	Comments
Passive Client	1	Only one panel can be connected to a single FieldServer serial port. Multiple networked panels can be supported through this connection to one panel.
Active Server (Simulate a Panel)	0	Not supported or documented.

2 Driver Scope of Supply

2.1 Supplied with this driver

FieldServer Technologies PART #	Description
8915-10	No specific cables are shipped with this driver. A generic RJ45 Ethernet cable is shipped with the hardware.
FS-8705-21	Driver Manual.

2.2 Provided by the Supplier of 3rd Party Equipment

2.2.1. Required 3rd Party Hardware

Part #	Description
MCU BOARD	FIELDSERVER CONNECTS TO TB8 OR TB9 ON THE MCU BOARD.

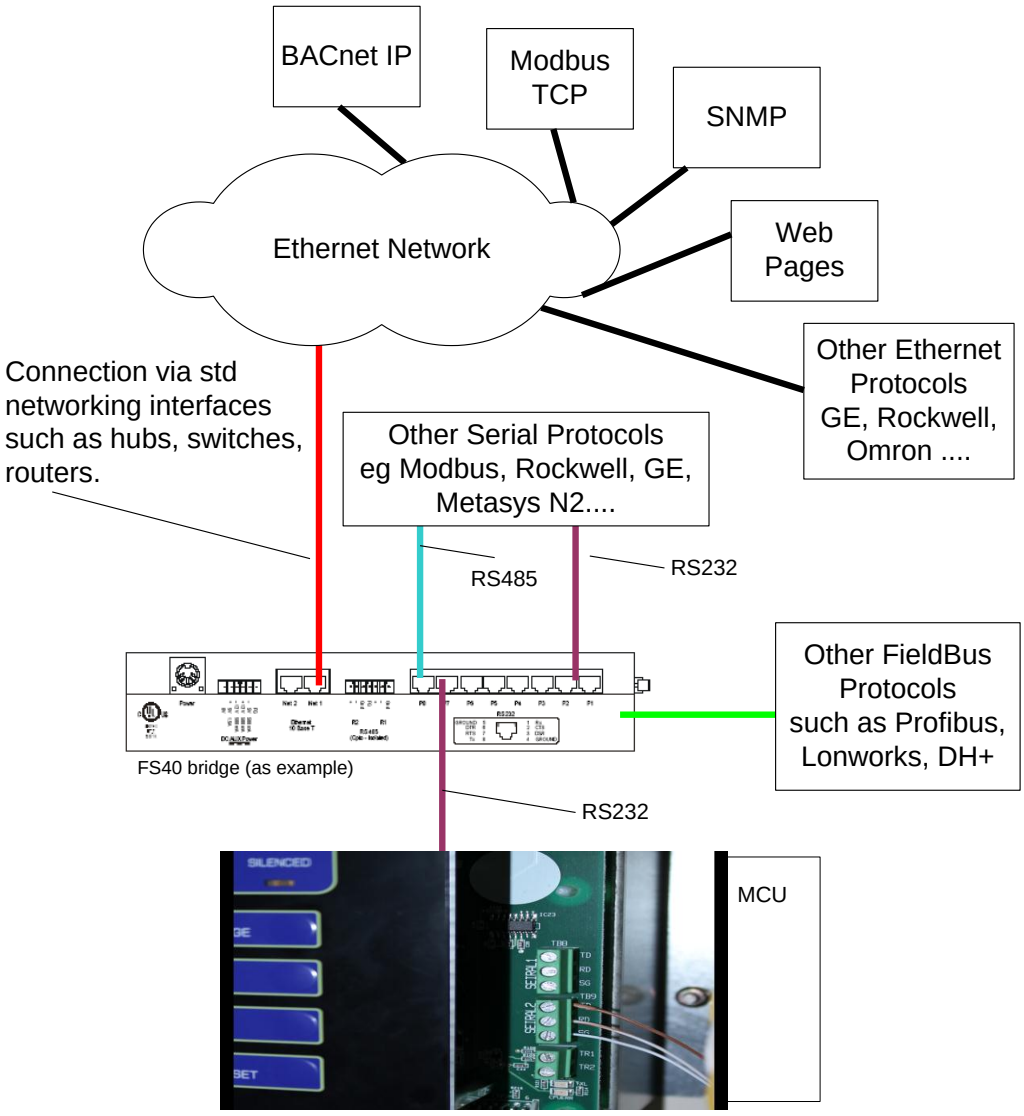
3 Hardware Connections

Multiple WorkStation protocols and connection supported. See list of FieldServer Drivers.

3.1 Hardware Connections

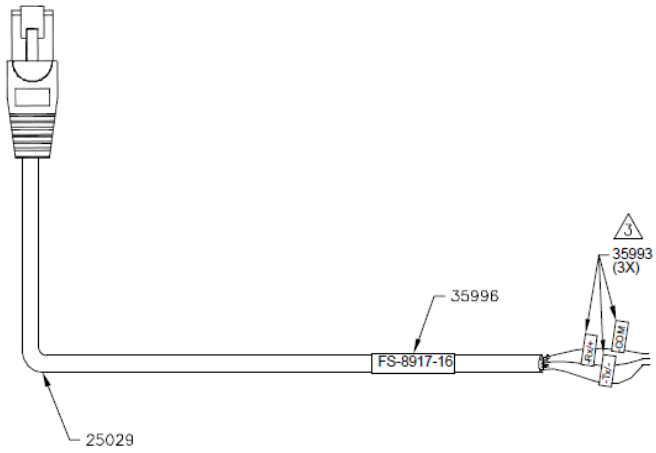
Connect the RS232 cable to TB8 or TB9 as shown.

3.2 Block Diagram



3.3 Recommended Cable Assembly

An RJ45 to Terminal cable can be constructed by cutting the end off a Ethernet patch cable (not a cross over cable) as shown below. This is used to connect the FieldServer RS-232 port (RJ-45) to the Integlex FACP. Note that some FieldServer models have terminal blocks for this connection instead of RJ-45, and for this a standard terminal to terminal wiring setup would apply.



NOTES: UNLESS OTHERWISE SPECIFIED:

1. CUT OFF ONE END OF CABLE.
2. CUT JACKET APPROX. 1.5". KEEP 3 WIRES: BRONW, BLUE/WHITE, ORANGE/WHIE. STRIP AND TIN THE ENDS.
3. LABEL WIRES:
   LABEL "FS-RX/+" FOR BROWN WIRE - RS45-01
   LABEL "FS-COM" FOR BLUE/WHITE WIRE - RS45-04
   LABEL "FS-TX/-" FOR ORANGE/WHITE WIRE - RS45-08

4 Configuring the FieldServer as a Integlex Passive Client

For a detailed discussion on FieldServer configuration, please refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (See ".csv" sample files provided with the FS).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with an Integlex system.

The configuration file tells the FieldServer about its interfaces, and the routing of data required. In order to enable the FieldServer for Integlex monitoring, the driver independent FieldServer buffers need to be declared in the "Data Arrays" section, the destination device addresses need to be declared in the "Client Side Nodes" section, and the data required from the servers needs to be mapped in the "Client Side Map Descriptors" section. Details on how to do this can be found below.

Note that in the tables, * indicates an optional parameter, with the bold legal value being the default.

4.1 Data Arrays

Section Title
Data_Arrays

Column Title	Function	Legal Values
Data_Array_Name	Provide name for Data Array	Up to 15 alphanumeric characters
Data_Array_Format	Provide data format. Each Data Array can only take on one format.	**Recommended**: Bit, UInt16, Also Supported: Float, Uint32, SInt16, Packed_Bit, Byte, Packed_Byte, Swapped_Byte
Data_Array_Length	Number of Data Objects. Must be larger than the data storage area required by the Map Descriptors for the data being placed in this array.	1-10,000

4.1.1 Data Arrays - Example

```
//   Data Arrays
Data_Arrays
Data_Array_Name,          Data_Format,      Data_Array_Length,

integlex_stats,           UNT16,            200
```

4.2 Client Side Connections

Create one connection for each Integlex serial port. Each connection can only be used to connect to a single Integlex interface/port.

Section Title
Connections

Column Title	Function	Legal Values
Port	Specify which port the device is connected to the FieldServer	P1-P8
Protocol	Specify protocol used	integlex
Baud*	Specify baud rate	Driver Supports : 110; 300; 600; 1200; 2400; 4800; **9600**; 19200; 28800; 38400; 57600 Baud *Integlex* supports: 9600
Data_Bits *	Specify parity	Driver Supports : 7,**8** *Integlex* supports: 7
Stop_Bits*	Specify data bits	Driver Supports : **1**,2 *Integlex* supports: 1
Parity *	Specify stop bits	Driver Supports : Odd, Even, **None** *Integlex* supports: Even

4.2.1 Client Side Connection Descriptions - Example

```
//      Client Side Connections


Connections

Port,           Baud,           Parity,     Protocol, Data_Bits , Stop_Bits

P1,             9600,           Even,       Integlex , 7        , 1
```

4.3 Client Side Nodes

Create one Node per FACP in the network only.

Section Title
Nodes

Column Title	Function	Legal Values
Node_Name	Provide name for node	Up to 32 alphanumeric characters **NB !** The name MUST correspond to the Node name configured in the Integlex system. If it does not then messages sent from networked nodes might be ignored.
Node_ID	Station address of physical server node This parameter is not used directly by the driver. We recommend that a unique Node ID's be given to each node.	0-258 Corresponds to the Node numbers of panels. Master node is typically zero.
Protocol	Specify protocol used	integlex

4.3.1 Client Side Nodes - Example

```
//   Client Side Nodes


Nodes
Node_Name,          Node_ID,        Protocol,       Connection
MainPanel,          0,              integlex ,      P1
```

4.4 Client Side Map Descriptors

4.4.1 FieldServer Related Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer	One of the Data Array names from "Data Array" section above
Data_Array_Offset	Starting location in Data Array	0 to maximum specified in "Data Array" section above
Function	Function of Client Map Descriptor..	Passive

4.4.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from	One of the node names specified in "Client Node Descriptor" above
Data_Type	This commonly used parameter is not used by this driver.	
Length	Length of Map Descriptor Reserves space in the Data Array.	1,2,3…. Set to the value of the maximum device number on the loop. For system troubles etc. set the length to 1
Address	This commonly used FieldServer parameter is not used by this protocol.	
Integlex_Function	Tell the driver what kind of event will be stored using this Map Descriptor. Eg. If you want 'Alert' events stored set the Integlex_Function =Alert	System_Trouble Panel_Trouble Alert Action Confirm Alarm

		Active Verify Supervisory No_Answer Comm_Fault Disable Type_Error Fault Data_Error Classification The following 2 items need to be used in a special way. Any_Alarm Any_Trbl
Integlex_Loop	If Integlex_Function = System_Trouble or Classification then this parameter should be specified with a dash or omitted. For other functions set the value of the parameter to the loop number whose events you want stored.	-, 1 , 2, ….

Integlex_Full_Summary_DA	A single device on a single loop can in be in multiple states simultaneously. For example, a Smoke detector can report an alert, an action or an alarm event. Normally these events are stored in different locations / Data Arrays. If you want a summary for a device then use this parameter. Read more in section 4.5.4 Summarizing Data	The Name of a Data Array defined in the Data Array section. Or a dash if not required.
Integlex_Summary_DA	A single device on a single loop can in be in multiple states simultaneously. For example, a Smoke detector can report an alert, an action or an alarm event. Normally these events are stored in different locations / Data Arrays. If you want a summary for a device then use this parameter. Read more in section 4.5.4 Summarizing Data	The Name of a Data Array defined in the Data Array section. Or a dash if not required.
Integlex_MultiAlarm_DA	A single device on a single loop can in be in multiple states simultaneously. Use this parameter to specify the name of a Data Array to be used to store a value which reports which is the most important activated Alarm state.	The Name of a Data Array defined in the Data Array section. Or a dash if not required.

Integlex_MultiTrbls_DA	Use this parameter to specify the name of a Data Array to be used to store a value which reports which Trouble is active.	The Name of a Data Array defined in the Data Array section. Or a dash if not required.

4.5 How Data is Stored / Summarizing Events

4.5.1 Classification

When a device on a loop reports an event, the driver learns the device classification (think of the classification as the device type. Eg. Analog Smoke Detector). The classification information is not cleared when a system reset occurs. Notes on the Classifications and a table of Classifications and their corresponding enumeration is provided in an appendix to this manual.

4.5.2 System/Panel Events

System/Panel events are those reported by the main/networked panels as opposed to a single addressable device. One Map Descriptor is required to store System events for main panel and one more for each networked panel in the system. The Node name in the configuration must correspond exactly to the panel name configured in the Integlex panel. System/Panel Trouble data is cleared when there is a System Reset.

The offset into the data array is used to indicate the event type. Eg. A Ground fault event will be indicated when offset 9 it set to 1. Standby Power Fault has offset 10 set to 1. The table below provides a full list of system troubles and the offset number. The table can be updated by the configuration engineer.

On the next pages a table is provided which can be used to identify the Data Array offset used by each type of System or Panel Trouble event.

Table of System/Panel Troubles and offset

System Trouble Event	Data Array Offset	System Trouble Event	Data Array Offset
"Loop 1 Loop-Back"	1	"OCU 2 Type Error"	26
"Loop 2 Loop-Back"	2	"OCU 3 Type Error"	27
"Loop 3 Loop-Back"	3	"OCU 4 Type Error"	28
"Loop 4 Loop-Back"	4	"OCU 5 Type Error"	29
"Loop 1 Short Circuit"	5	"OCU 6 Type Error"	30
"Loop 2 Short Circuit"	6	"OCU 7 Type Error"	31
"Loop 3 Short Circuit"	7	"OCU 8 Type Error"	32
"Loop 4 Short Circuit"	8	"Remote Annunciator #01 Trouble"	33
"Ground Fault"	9	"Remote Annunciator #02 Trouble"	34
"Standby Power Fault"	10	"Remote Annunciator #03 Trouble"	35
"Connection Error"	11	"Remote Annunciator #04 Trouble"	36
"LCD Trouble"	12	"Remote Annunciator #05 Trouble"	37
"SCU 1 Trouble"	13	"Remote Annunciator #06 Trouble"	38
"SCU 2 Trouble"	14	"Remote Annunciator #07 Trouble"	39
"SCU 3 Trouble"	15	"Remote Annunciator #08 Trouble"	40
"NIU Trouble"	16	"Remote Annunciator #09 Trouble"	41
"OCU 1 Trouble"	17	"Remote Annunciator #10 Trouble"	42
"OCU 2 Trouble"	18	"Remote Annunciator #11 Trouble"	43
"OCU 3 Trouble"	19	"Remote Annunciator #12 Trouble"	44
"OCU 4 Trouble"	20	"Remote Annunciator #13 Trouble"	45
"OCU 5 Trouble"	21	"Remote Annunciator #14 Trouble"	46
"OCU 6 Trouble"	22	"Remote Annunciator #15 Trouble"	47
"OCU 7 Trouble"	23	"Remote Annunciator #16 Trouble"	48
"OCU 8 Trouble"	24	"Remote Annunciator #17 Trouble"	49
"OCU 1 Type Error"	25	"Remote Annunciator #18 Trouble"	50

Continued on the next page

Table of System/Panel Troubles and offset

System Trouble Event	Data Array Offset	System Trouble Event	Data Array Offset
"Remote Annunciator #19 Trouble"	51	"Configuration Data Upload"	76
"Remote Annunciator #20 Trouble"	52	"Operating Program Download"	77
"Remote Annunciator #21 Trouble"	53	"Event Log Data Upload"	78
"Remote Annunciator #22 Trouble"	54	"Maintenance List Data Upload"	79
"Remote Annunciator #23 Trouble"	55	"Node #01 Network Failure"	80
"Remote Annunciator #24 Trouble"	56	"Node #02 Network Failure"	81
"Remote Annunciator #25 Trouble"	57	"Node #03 Network Failure"	82
"Remote Annunciator #26 Trouble"	58	"Node #04 Network Failure"	83
"Remote Annunciator #27 Trouble"	59	"Node #05 Network Failure"	84
"Remote Annunciator #28 Trouble"	60	"Node #06 Network Failure"	85
"Remote Annunciator #29 Trouble"	61	"Node #07 Network Failure"	86
"Remote Annunciator #30 Trouble"	62	"Node #08 Network Failure"	87
"CIM Trouble"	63	"Node #09 Network Failure"	88
"Loop 1 Power Fault"	64	"Node #10 Network Failure"	89
"Loop 2 Power Fault"	65	"Node #11 Network Failure"	90
"Loop 3 Power Fault"	66	"Node #12 Network Failure"	91
"Loop 4 Power Fault"	67	"Node #13 Network Failure"	92
"Trouble Input"	68	"Node #14 Network Failure"	93
"Drill"	69	"Node #15 Network Failure"	94
"Silence"	70	"Node #16 Network Failure"	95
"Network Ground Fault"	71	"Node #17 Network Failure"	96
"Network Failure Port A"	72	"Node #18 Network Failure"	97
"Network Failure Port B"	73	"Node #19 Network Failure"	98
"Duplicate Node Numbers"	74	"Node #20 Network Failure"	99
"Configuration Data Download"	75	"Node #21 Network Failure"	100

Continued on the next page

Table of System/Panel Troubles and offset

System Trouble Event	Data Array Offset	System Trouble Event	Data Array Offset
"Node #22 Network Failure"	101	"Node #57 Network Failure"	126
"Node #23 Network Failure"	102	"Node #58 Network Failure"	127
"Node #24 Network Failure"	103	"Node #59 Network Failure"	128
"Node #25 Network Failure"	104	"Node #60 Network Failure"	129
"Node #26 Network Failure"	105	"Node #61 Network Failure"	130
"Node #27 Network Failure"	106	"Node #62 Network Failure"	131
"Node #28 Network Failure"	107	"Node #63 Network Failure"	132
"Node #29 Network Failure"	108	"Node #64 Network Failure"	133
"Node #30 Network Failure"	109	"Main Power Fault"	134
"Node #41 Network Failure"	110	"Trouble"	135
"Node #42 Network Failure"	111	""	999
"Node #43 Network Failure"	112		
"Node #44 Network Failure"	113		
"Node #45 Network Failure"	114		
"Node #46 Network Failure"	115		
"Node #47 Network Failure"	116		
"Node #48 Network Failure"	117		
"Node #49 Network Failure"	118		
"Node #50 Network Failure"	119		
"Node #51 Network Failure"	120		
"Node #52 Network Failure"	121		
"Node #53 Network Failure"	122		
"Node #54 Network Failure"	123		
"Node #55 Network Failure"	124		
"Node #56 Network Failure"	125		

4.5.3 Device / Module Events

A device event, for the purposes of this manual, is one reported by an addressable device on a loop.

The driver is configured by making a map descriptor for each loop on each panel. For each type of event you are interested in you make a map descriptor. The offset into the data array is the device number.

Eg. If you are using a Data Array called **DA_N1L1_Alert** to store Alert events for Loop 1 on panel 1, then when device 1 reports an alert the driver will set DA_N1L1_Alert[1]=1. If device 10 reports an alert event, then DA_N1L1_Alert[10]=1. When the event clears then the value in the Data Array is set to zero.

Eg. If you are using a Data Array called **DA_N1L1_Alarms** to store Alarm events for Loop 1 on panel 1, then when device 1 report an Alarm the driver will set **DA_N1L1_Alarms** [1]=1. If device 10 reports an Alarm event then **DA_N1L1_Alarms** [10]=1. When the event clears then the value in the Data Array is set to zero.

4.5.4 Summarizing Data

A single device on a single loop can in be in multiple states simultaneously. For example, a Smoke detector can report an *alert*, an *action* or an *alarm* event. Normally these events are stored in different locations / Data Arrays.

What happens if you don't care if its in alarm or alert, what you care about is the abnormal state? Then you should use the Summary system.

To do this, allocate a Data Array to store the summary. The Array Data type should be UINT16 – a 16 bit word. The driver allocates a bit for each state. Thus if a device is in multiple states then multiple bits will be set. You simply check the word to see if it is zero (no bits set meaning all normal) or non-zero meaning at least one active state. You could go further and extract the bits

to learn the specific state. (See the FieldServer Bridge Configuration Manual for notes on how to use the Bit_Extract function.)

There are 2 summary systems. The one system (Full Summary) has a unique bit number for all possible events making it useful to make a summary that contains active states and troubles. The second system allocates separate summaries for actives and troubles so you need to use two different data arrays.

4.5.4.1 Full Summary

If you specify the parameter **Integlex_Full_Summary_DA** then when the driver stores the event it will also update this summary Data Array. Again, the Data Array offset number is equal to the device number. In this case, the driver does more than store a 1 or a zero. It set the bit in the number to 1 or zero. Thus the value changes as the events go on/off. The following table shows the relationship between bit number and event type.

Activation Events			**Trouble Events**	
Event Type	**Bit Number**		**Event Type**	**Bit Number**
Alert	0		No_Answer	9
Action	1		Comm_Fault	10
Confirm	2		Disable	11
Alarm	3		Type_Error	12
Active (formerly AlarmF1)	4		Fault	13
Not Used (formerly AlarmF2)	5		Data_Error	14
Verify	6			
Supervisory	7			

While it may make sense to use Data Arrays whose format=BIT for the events, for the summary Data Array it makes sense to use a UINT16 .

4.5.4.2 Summary (Not Full)

If you specify the parameter **Integlex_Summary_DA** then when the driver stores the event it will also update this summary Data Array. Again, the Data Array offset number is equal to the device number. In this case, the driver does more than store a 1 or a zero. It set the bit in the number to 1 or zero. Thus the value changes as the events go on/off. The following table shows the relationship between bit number and event type.

Note how some bit numbers are used twice. That means you cannot use this summary method to summarize troubles and actives together (use the full summary). Using this method you should use one Data Array for Actives and a different one for Troubles (an example is provided below)

Activation Events			**Trouble Events**	
Event Type	**Bit Number**		**Event Type**	**Bit Number**
Alert	0		No_Answer	0
Action	1		Comm_Fault	1
Confirm	2		Disable	2
Alarm	3		Type_Error	3
Active (formerly AlarmF1)	4		Fault	4
Not Used (formerly AlarmF2)	5		Data_Error	5
Verify	6			
Supervisory	7			

While it may make sense to use Data Arrays whose format=BIT for the events, for the summary Data Array it makes sense to use a UINT16 or Byte .

Example:

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop        ,Integlex_Summary_DA

StoreClassific      ,N1_LOOP1_class   ,0                ,Server   ,Node1     ,Classification    ,1                   ,-                 ,

StoreAlerts         ,N1_LOOP1         ,0                ,Server   ,Node1     ,Any_Alarm         ,1                   ,N1_LOOP1_ACTIVES  ,

StoreNo_Asnwer      ,N1_LOOP1         ,0                ,Server   ,Node1     ,Any_Trble         ,1                   ,N1_LOOP1_TRBLS    ,
```

4.5.5 MultiState Data

The driver presents a system for building a MultiState value for device / module states.

If a device is in an alarm state, the driver provides a number whose value reports the most important / serious of the possible active states. For example, if a device is in Alert and Alarm States simultaneously then the driver will report the Alarm state since it is considered a higher priority.

If a device is a trouble state, then the driver provides a number whose value reports which trouble state is active (if only one state is active) or which reports that multiple states are active when more than one trouble is active.

4.5.5.1 MultiState Alarms

The driver can set a Data Array location to one of the tabulated values below based on the current activated *Alarm* states. If any two or more states are active at the same time, the driver sets the value to the one with the highest value. Eg. If Alert and Alarm States for a device are active then the Multistate Value will be set to 4 because Alarm (4) has a higher value than Alert (1).

MultiState Description	Value
All Normal	0
Alert	1
Action	2
Confirm	3
Alarm	4
Active	5
Not Used	6
Verify	7
Supervisory	8

This system requires that you implement the summary described in section 4.6.4 Summary (Not Full) or it will not operate correctly. See section for an example.

4.5.5.2 MultiState Troubles

The driver can set a Data Array location to one of the tabulated values below based on the current activated Trouble states. If any two or more states are active at the same time, the driver sets the value to the one with the highest value. Eg. If Alert and Alarm States for a device are active then the Multistate Value will be set to 4 because Alarm (4) has a higher value than Alert (1).

MultiState Description	Value
No Troubles	0
No_Answer	1
Comm_Fault	2
Disable	3
Type_Error	4
Fault	5
Data_Error	6
Multiple Trouble States	15

This system requires that you implement the summary described in section 4.6.4 Summary (Not Full) or it will not operate correctly. See section for an example.

4.5.6 On Reset

Device and summary data is cleared when there is a System Reset

4.6 Some More Examples

4.6.1 Map Descriptor Example 1 – Store System Troubles

System Troubles are those reported by the Main MCU. Troubles reported by networked panels/nodes are called Panel Troubles by this driver. Therefore you can only have one Map Descriptor capturing System Troubles but you can have multiple Map Descriptors capturing Panel Troubles. If a System Trouble Event is reported by 'Node1' then the Data Array N1_SysTrb will be updated. The offset into the data array indicates the type of system trouble. It is important that in the Node section of the config, that the main panel is allocated a Node_ID which corresponds to the Main Panel number and that the Node Name is correctly spelled.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop         ,Integlex_Summary_DA

StoreSysTroubles    ,N1_SYSTRB        ,0                 ,Server   ,Node1     ,System_Trouble    ,-                    ,-                 ,
```

Descriptive only. Not used.

Data Array and offset where data will be stored.

Wait passively for events to be reported.

Node Name of a Node defined in the Nodes section. Node Number must correspond to the Integlex panel number. Name must match too.

Store System Troubles (for this node)

Not required when storing System Troubles

Not required when storing System Troubles

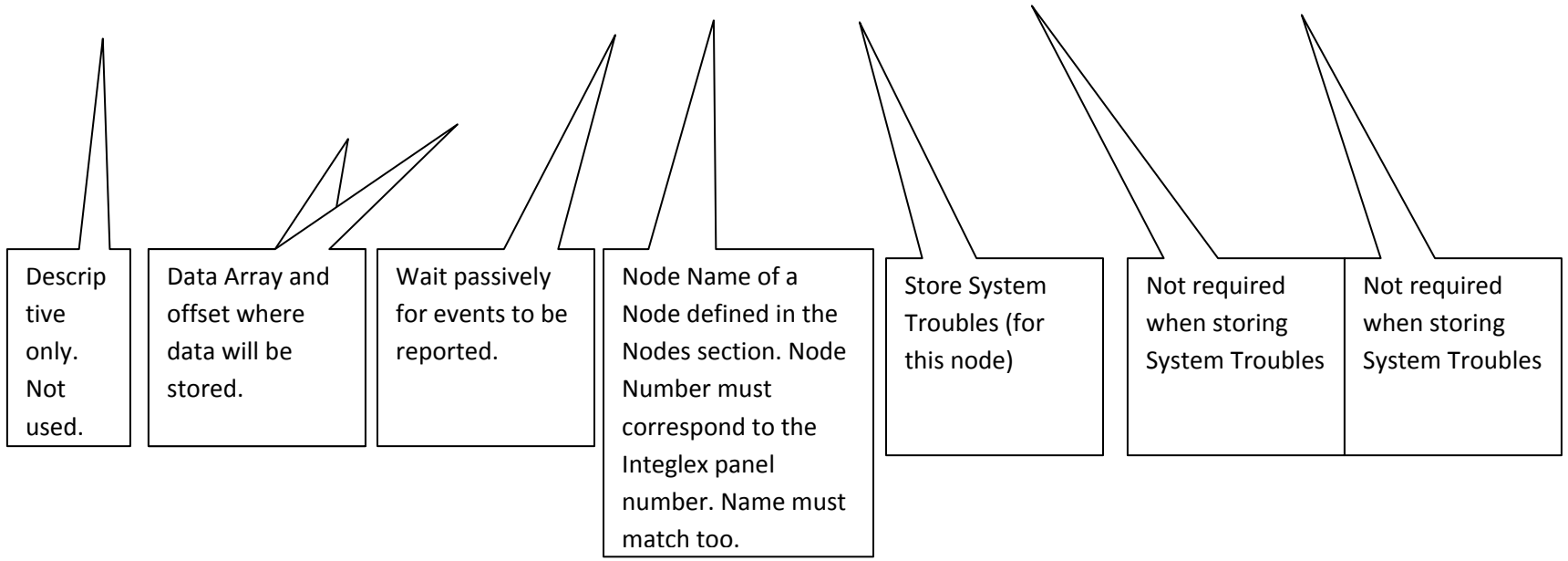
4.6.2 Map Descriptor Example 2 – Store Panel Troubles

System Troubles are those reported by the Main MCU. Troubles reported by networked panels/nodes are called Panel Troubles by this driver. You can have multiple Map Descriptors capturing Panel Troubles. If a Panel Trouble Event is reported by 'Node2' then the Data Array N2_SysTrb will be updated. The offset into the data array indicates the type of system trouble. It is important that in the Node section of the config, that the main panel is allocated a Node_ID which corresponds to the Networked Panel number **and that the Node Name is correctly spelled**.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop         ,Integlex_Summary_DA

StoreSysTroubles    ,N1_SYSTRB        ,0                 ,Server   ,Node1     ,Panel_Trouble     ,-                    ,-                 ,
```



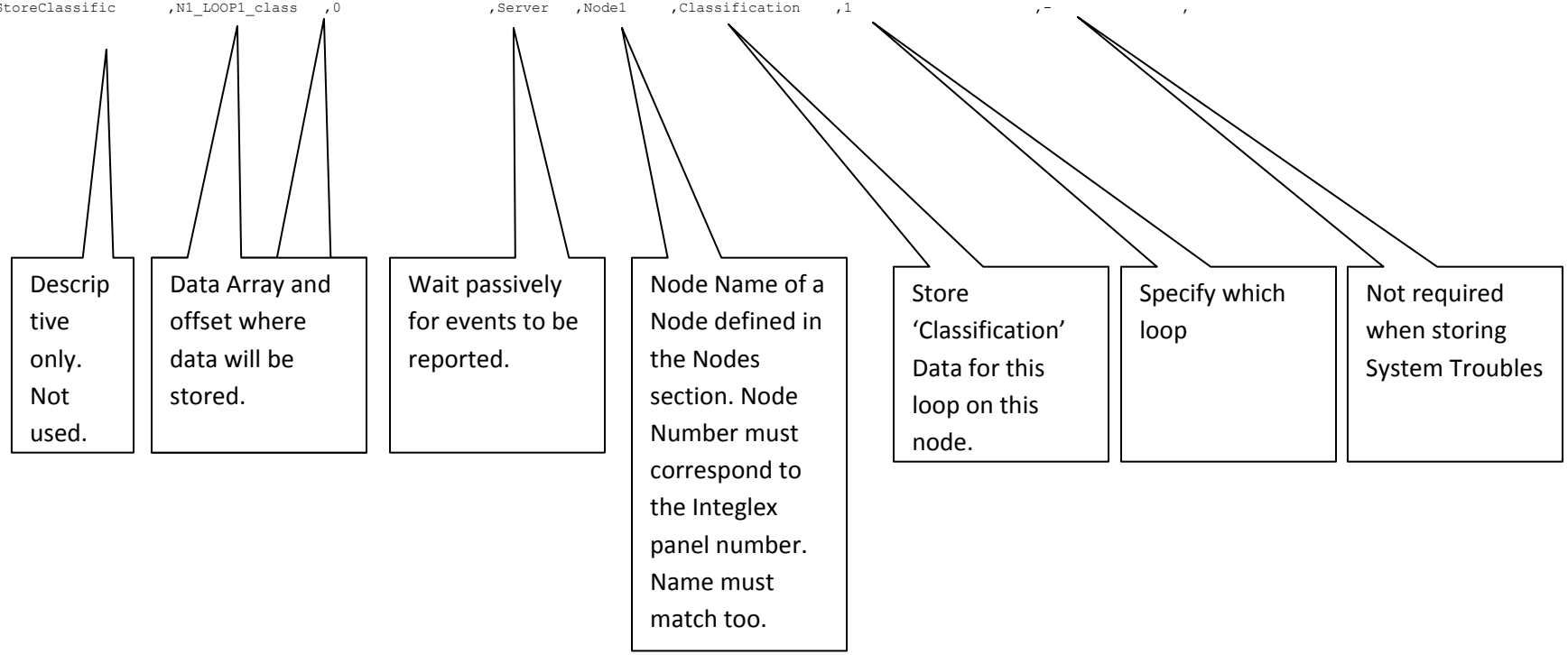
4.6.3 Map Descriptor Example 3 – Store Classifications

You can have the driver store a number corresponding to the type or classification of an addressable point on a loop (eg a device) using one Map Descriptor per loop per panel.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop        ,Integlex_Summary_DA

StoreClassific      ,N1_LOOP1_class   ,0                ,Server   ,Node1     ,Classification    ,1                  ,-                 ,
```



4.6.4 Map Descriptor Example 4 – Store Device Event Data

In this example Alert Events for Loop 2 on Node 1 are processed. Each time one of these events reports the driver looks for a Map Descriptor whose Integlex_Function=Alert and whose Loop Number and node match. If one is found, the driver looks up the specificed Data Array and offset and sets the value to 1 (On) or zero(off). The offset number equals the device number (device's address on the loop). So if Loop 2 Device 10 reports an Alert On then expect Data Array N1_LOOP2_ALERT[10] to be 1.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop         ,Integlex_Summary_DA

StoreAlerts         ,N1_LOOP2_ALERT   ,0                 ,Server   ,Node1     ,Alert             ,2                    ,N1_LOOP2_SUMM   ,
```

4.6.5 Map Descriptor Example 5 – Store Device Event Data

Like the previous example but this time, we have defined one map descriptor for each event type.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop         ,Integlex_Full_Summary_DA

StoreAlerts         ,N1_LOOP2_ALERT   ,0                 ,Server   ,Node1     ,Alert            ,2                   ,N1_LOOP2_SUMM   ,

StoreActions        ,N1_LOOP2_ACTION  ,0                 ,Server   ,Node1     ,Action           ,2                   ,N1_LOOP2_SUMM   ,

StoreConfirms       ,N1_LOOP2_CONFIR  ,0                 ,Server   ,Node1     ,Confirm          ,2                   ,N1_LOOP2_SUMM   ,

StoreAlarms         ,N1_LOOP2_ALARM   ,0                 ,Server   ,Node1     ,Alarm            ,2                   ,N1_LOOP2_SUMM   ,

StoreVerify         ,N1_LOOP2_VERIFY  ,0                 ,Server   ,Node1     ,Verify           ,2                   ,N1_LOOP2_SUMM   ,

StoreSupervisory    ,N1_LOOP2_SUPER   ,0                 ,Server   ,Node1     ,Supervisory      ,2                   ,N1_LOOP2_SUMM   ,

StoreNo_Asnwer      ,N1_LOOP2_NO_ANS  ,0                 ,Server   ,Node1     ,No_Answer        ,2                   ,N1_LOOP2_SUMM   ,

StoreCommFault      ,N1_LOOP2_COMMFL  ,0                 ,Server   ,Node1     ,Comm_Fault       ,2                   ,N1_LOOP2_SUMM   ,

StoreDisable        ,N1_LOOP2_DISABL  ,0                 ,Server   ,Node1     ,Disable          ,2                   ,N1_LOOP2_SUMM   ,

StoreTypeErrors     ,N1_LOOP2_TYPERR  ,0                 ,Server   ,Node1     ,Type_Error       ,2                   ,N1_LOOP2_SUMM   ,

StoreFaults         ,N1_LOOP2_FAULT   ,0                 ,Server   ,Node1     ,Fault            ,2                   ,N1_LOOP2_SUMM   ,

StoreDataErrors     ,N1_LOOP2_DATAER  ,0                 ,Server   ,Node1     ,Data_Error       ,2                   ,N1_LOOP2_SUMM   ,
```

4.6.6 Map Descriptor Example 6 – Summarizing Data

In this example 2 different summary Data Arrays are specified. One summarizes activation states and the other summarizes trouble states. For example: If an ALARM event occurs on N1-1002 then the driver updates N1_LOOP1_ALERT [2] by setting its value to 1. It also goes to the Data Array N1_LOOP1_ALARM[2] and sets bit 3 in the N1_LOOP1_ACTIVES[2]. Offset 2 applies because the device address is 002. Bit 3 was modified in the existing value found N1_LOOP1_ACTIVES[2] because when we read section 4.5.4.2 Summary (Not Full) we learn that Alarms affect Bit 3. So if the previous value in N1_LOOP1_ACTIVES[2] was zero then after the Alarm the value will be 8 (bit 3 set).

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop         ,Integlex_Summary_DA

StoreClassific      ,N1_LOOP1_class   ,0                ,Server   ,Node1     ,Classification    ,1                   ,-                ,

StoreAlerts         ,N1_LOOP1_ALERT   ,0                ,Server   ,Node1     ,Alert             ,1                   ,N1_LOOP1_ACTIVES   ,

StoreActions        ,N1_LOOP1_ACTION  ,0                ,Server   ,Node1     ,Action            ,1                   ,N1_LOOP1_ACTIVES   ,

StoreConfirms       ,N1_LOOP1_CONFIR  ,0                ,Server   ,Node1     ,Confirm           ,1                   ,N1_LOOP1_ACTIVES   ,

StoreAlarms         ,N1_LOOP1_ALARM   ,0                ,Server   ,Node1     ,Alarm             ,1                   ,N1_LOOP1_ACTIVES   ,

StoreVerify         ,N1_LOOP1_VERIFY  ,0                ,Server   ,Node1     ,Verify            ,1                   ,N1_LOOP1_ACTIVES   ,

StoreSupervisory    ,N1_LOOP1_SUPER   ,0                ,Server   ,Node1     ,Supervisory       ,1                   ,N1_LOOP1_ACTIVES   ,

StoreNo_Asnwer      ,N1_LOOP1_NO_ANS  ,0                ,Server   ,Node1     ,No_Answer         ,1                   ,N1_LOOP1_TRBLS     ,

StoreCommFault      ,N1_LOOP1_COMMFL  ,0                ,Server   ,Node1     ,Comm_Fault        ,1                   ,N1_LOOP1_TRBLS   ,

StoreDisable        ,N1_LOOP1_DISABL  ,0                ,Server   ,Node1     ,Disable           ,1                   ,N1_LOOP1_TRBLS   ,

StoreTypeErrors     ,N1_LOOP1_TYPERR  ,0                ,Server   ,Node1     ,Type_Error        ,1                   ,N1_LOOP1_TRBLS   ,

StoreFaults         ,N1_LOOP1_FAULT   ,0                ,Server   ,Node1     ,Fault             ,1                   ,N1_LOOP1_TRBLS   ,

StoreDataErrors     ,N1_LOOP1_DATAER  ,0                ,Server   ,Node1     ,Data_Error        ,1                   ,N1_LOOP1_TRBLS   ,
```

4.6.7 Map Descriptor Example 7 – Summarizing Data (Full)

This is a full summary. The same DA is used to summarize troubles and active states. (DA_LOOP1_SUMMARY). On the next page we trace the value of the summary as various events occur.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop        ,Integlex_FULL_Summary_DA

StoreClassific      ,N1_LOOP1_class   ,0                ,Server   ,Node1     ,Classification    ,1                    ,-                 ,

StoreAlerts         ,N1_LOOP1_ALERT   ,0                ,Server   ,Node1     ,Alert             ,1                    ,N1_LOOP1_SUMMARY   ,

StoreActions        ,N1_LOOP1_ACTION  ,0                ,Server   ,Node1     ,Action            ,1                    ,N1_LOOP1_SUMMARY   ,

StoreConfirms       ,N1_LOOP1_CONFIR  ,0                ,Server   ,Node1     ,Confirm           ,1                    ,N1_LOOP1_SUMMARY      ,

StoreAlarms         ,N1_LOOP1_ALARM   ,0                ,Server   ,Node1     ,Alarm             ,1                    ,N1_LOOP1_SUMMARY,

StoreVerify         ,N1_LOOP1_VERIFY  ,0                ,Server   ,Node1     ,Verify            ,1                    ,N1_LOOP1_SUMMARY,

StoreSupervisory    ,N1_LOOP1_SUPER   ,0                ,Server   ,Node1     ,Supervisory       ,1                    ,N1_LOOP1_SUMMARY,

StoreNo_Asnwer      ,N1_LOOP1_NO_ANS  ,0                ,Server   ,Node1     ,No_Answer         ,1                    ,N1_LOOP1_SUMMARY,

StoreCommFault      ,N1_LOOP1_COMMFL  ,0                ,Server   ,Node1     ,Comm_Fault        ,1                    ,N1_LOOP1_SUMMARY,

StoreDisable        ,N1_LOOP1_DISABL  ,0                ,Server   ,Node1     ,Disable           ,1                    ,N1_LOOP1_SUMMARY,

StoreTypeErrors     ,N1_LOOP1_TYPERR  ,0                ,Server   ,Node1     ,Type_Error        ,1                    ,N1_LOOP1_SUMMARY,

StoreFaults         ,N1_LOOP1_FAULT   ,0                ,Server   ,Node1     ,Fault             ,1                    ,N1_LOOP1_SUMMARY,

StoreDataErrors     ,N1_LOOP1_DATAER  ,0                ,Server   ,Node1     ,Data_Error        ,1                    ,N1_LOOP1_SUMMARY,
```

(Example notes continue on next page)

Assume all these events occur for a device at N01-1002. Then the Data Array offset affected is offset 2 (corresponds to device address).

Assume that DA_LOOP1_SUMMARY[2] = 0 at the start of the events

ALERT on event.; Bit 0: Value of DA_LOOP1_SUMMARY[2] before event = 0 Value after update = 1 (Binary=1)

ACTION event.; Bit1 : Value of DA_LOOP1_SUMMARY[2] before event = 1 Value after update = 3 (Binary=11)

ALARM on event.; Bit 3: Value of DA_LOOP1_SUMMARY[2] before event = 3 Value after update = 11 (Binary=1011)

NO ANSWER on event.; Bit 9: Value of DA_LOOP1_SUMMARY[2] before event =11 Value after update = 267 (Binary= 100001011)

4.6.8 Map Descriptor Example 8 – MultiState Data

This example uses example 6 and expands on it. The Data Array N1_L1_MultiALM will contain a number representing the Alarm state as a MultiState value. The offset into the Data Array is equal to the device/module reporting the events. The Data Array N1_L1_MultiTRB will contain a number reporting the single active trouble event or that multiple events are active at the same time.

Notes on the next page provide some information on how the value would change for a series of events.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array , ... ,Integlex_Function ,Integlex_Loop ,Integlex_Summary ,Integlex_MultiTrbls,Integlex_MultiAlarm
                    ,                 ,_Offset    ,     ,                  ,              ,_DA               ,_DA                ,_DA
StoreAlerts         ,N1_LOOP1_ALERT   ,0          , ... ,Alert             ,1             ,N1_LOOP1_ACTIVE   ,-                  ,N1_L1_MultiALM
StoreActions        ,N1_LOOP1_ACTION  ,0          , ... ,Action            ,1             ,N1_LOOP1_ACTIVE   ,-                  ,N1_L1_MultiALM
StoreConfirms       ,N1_LOOP1_CONFIR  ,0          , ... ,Confirm           ,1             ,N1_LOOP1_ACTIVE   ,-                  ,N1_L1_MultiALM
StoreAlarms         ,N1_LOOP1_ALARM   ,0          , ... ,Alarm             ,1             ,N1_LOOP1_ACTIVE   ,-                  ,N1_L1_MultiALM
StoreVerify         ,N1_LOOP1_VERIFY  ,0          , ... ,Verify            ,1             ,N1_LOOP1_ACTIVE   ,-                  ,N1_L1_MultiALM
StoreSupervisory    ,N1_LOOP1_SUPER   ,0          , ... ,Supervisory       ,1             ,N1_LOOP1_ACTIVE   ,-                  ,N1_L1_MultiALM
StoreNo_Asnwer      ,N1_LOOP1_NO_ANS  ,0          , ... ,No_Answer         ,1             ,N1_LOOP1_TRBLS    ,N1_L1_MultiTRB     ,-
StoreCommFault      ,N1_LOOP1_COMMFL  ,0          , ... ,Comm_Fault        ,1             ,N1_LOOP1_TRBLS    ,N1_L1_MultiTRB     ,-
StoreDisable        ,N1_LOOP1_DISABL  ,0          , ... ,Disable           ,1             ,N1_LOOP1_TRBLS    ,N1_L1_MultiTRB     ,-
StoreTypeErrors     ,N1_LOOP1_TYPERR  ,0          , ... ,Type_Error        ,1             ,N1_LOOP1_TRBLS    ,N1_L1_MultiTRB     ,-
StoreFaults         ,N1_LOOP1_FAULT   ,0          , ... ,Fault             ,1             ,N1_LOOP1_TRBLS    ,N1_L1_MultiTRB     ,-
StoreDataErrors     ,N1_LOOP1_DATAER  ,0          , ... ,Data_Error        ,1             ,N1_LOOP1_TRBLS    ,N1_L1_MultiTRB     ,-
```

Assume all these events occur for a device at N01-1002. Then the Data Array offset affected is offset 2 (corresponds to device address).

Assume that a system reset was performed and that there are no active events before the ones listed below.

ALERT on event.; Bit 0: Value of N1_LOOP1_ACTIVE[2] before event = 0 Value after update = 1 (Binary=1)

```
N1_L1_MultiALM[2] = 1
```

ACTION event.; Bit1 : Value of N1_LOOP1_ACTIVE[2] before event = 1 Value after update = 3 (Binary=11)

```
N1_L1_MultiALM[2] = 2 (Action(2) is more important than Alert(1))
```

ALARM on event.; Bit 3: Value of N1_LOOP1_ACTIVE[2] before event = 3 Value after update = 11 (Binary=1011)

```
N1_L1_MultiALM[2] = 4 (Alarm(4) is more important than Action(2) which is more important than Alert(1))
```

NO ANSWER on event.; Bit 0: Value of N1_LOOP1_TRBLS[2] before event =0 Value after update = 1 (Binary= 1)

```
N1_L1_MultiTRB[2] = 1 (No Asnwer) (See table in section 4.5.5.2 MultiState Troubles )
```

FAULT on event.; Bit 4: Value of N1_LOOP1_TRBLS[2] before event =1 Value after update = 17 (Binary= 10001)

```
N1_L1_MultiTRB[2] = 15 (Multiple Trouble active)
```

NO ANSWER **off** event.; Bit 0: Value of N1_LOOP1_TRBLS[2] before event =17 Value after update =161 (Binary= 10000)

```
N1_L1_MultiTRB[2] = 5 (Fault) (See table in section 4.5.5.2 MultiState Troubles )
```

4.6.9 Map Descriptor Example 9 – Simplified Config for MultiState Data

In this example a We have used the functions ‘Any_Alarm’ and ‘Any_Trbl’ . When this is done some information is lost – the driver does not store the individual state bits for each device. Rather, it simply maintains the summary and multistate. When an event occurs the driver decides if it’s an activation event or a trouble event. It looks for a place to store. If it’s an activation it looks for a MD whose function is specific to the event and/or it looks for a MD whose function=Any…. Alarm/Trbl and uses the summary and multistate arrays specified.

```
Map_Descriptors

Map_Descriptor_Name ,Data_Array_Name  ,Data_Array_Offset ,Function ,Node_Name ,Integlex_Function ,Integlex_Loop ,Integlex_Summary_DA ,Integlex_MultiTrbls_DA,Integlex_MultiAlarm_DA

StoreActivations    ,N1_DUMMY         ,0                ,Server   ,Node1     ,Any_alarm         ,1             ,N1_LOOP1_ACTIVE     ,-                    ,N1_L1_MultiALM

StoreTroubles       ,N1_DUMMY         ,0                ,Server   ,Node1     ,Any_trbl          ,1             ,N1_LOOP1_TRBLS      ,N1_L1_MultiTRB       ,-
```

This page left blank.

5 Configuring the FieldServer as a Integlex FACP Server

This driver cannot be used to emulate an Integlex Panel

Appendix 1 – Advanced Topics

Appendix 1.1. System Reset / Synch Panel and Gateway

A consequence of the fact that this is a passive client driver, is that the FieldServer must be synchronized with the panel by resetting the panel and then restarting the FieldServer to reset the data.

Ensure all points are normal and there are no active conditions. Perform a system Reset. This will result in a synch between panel and gateway. Even if you do not clear all active conditions 1st this method may be effective since, the panel should re-send status messages reporting all active conditions after the reset. You should test to ensure this method works for you before using it.

Appendix 1.2. Loading New Classifications

A driver contains a defualt list of classfications. The list may be updated. The driver looks for and if found, loads a file called classi.ini. In this case it prints a message and lsists all the classifications that were loaded. When classi.ini is loaded the default table is ignored. A maximum of 100 can be loaded.

Classification	#		Classification	#
Analog Smk Detector	1		Action Input	27
F/T Heat Detector	2		Alert Input	28
Comb. Heat Detector	3		Control Output	29
LaserCOMPACT LP	4		Bell Circuit	30
Photo Smk Detector	5		Horn Circuit	31
Ion Smoke Detector	6		Strobe Circuit	32
Heat Detector	7		Audible Circuit	33
Smoke/Heat Detector	8		Speaker Circuit	34
Leak Detector	9		Release End Bell	35
Pull Station	10		Release Circuit	36
Monitor Module	11		Release Audible	37
Waterflow Monitor	12		Telephone	38
Manual Release	13		Relay	39
Sprinkler System	14		Power Shutdown	40
Alarm Input	15		HVAC Shutdown	41
Second Shot	16		FAN Shutdown	42
Supervisory	17		Release Form C	43
Tamper	18		Fire Alarm Status	44
Abort Switch	19		Trouble Status	45
Trouble Input	20		Supervisory Status	46
Fault Input	21		Disable Status	47
Leak Detector Fault	22		System Reset Status	48
PAS Inhibit	23		Silence Status	49
Hazard Alert	24		Drill Status	50
Silent Input	25		Main Power Fault	51
ZOCU Switch	26		Standby Power Fault	52

Appendix 1.3. Loading New System/Panel Troubles Event Types

The driver contains a list of default trouble strings. This list can be updated. . The driver looks for and if found, loads a file called systrb.ini. In this case it prints a message and lists all the trouble strings that were loaded. When systrb.ini is loaded the default table is ignored. A maximum of 200 System Trouble Event types can be loaded.

Default Values are tabulated below:-

System/Panel Trouble Type	Enumeration		System/Panel Trouble Type	Enumeration
Loop 1 Loop-Back	1		OCU 2 Type Error	26
Loop 2 Loop-Back	2		OCU 3 Type Error	27
Loop 3 Loop-Back	3		OCU 4 Type Error	28
Loop 4 Loop-Back	4		OCU 5 Type Error	29
Loop 1 Short Circuit	5		OCU 6 Type Error	30
Loop 2 Short Circuit	6		OCU 7 Type Error	31
Loop 3 Short Circuit	7		OCU 8 Type Error	32
Loop 4 Short Circuit	8		Remote Annunciator #01 Trouble	33
Ground Fault	9		Remote Annunciator #02 Trouble	34
Standby Power Fault	10		Remote Annunciator #03 Trouble	35
Connection Error	11		Remote Annunciator #04 Trouble	36
LCD Trouble	12		Remote Annunciator #05 Trouble	37
SCU 1 Trouble	13		Remote Annunciator #06 Trouble	38
SCU 2 Trouble	14		Remote Annunciator #07 Trouble	39
SCU 3 Trouble	15		Remote Annunciator #08 Trouble	40
NIU Trouble	16		Remote Annunciator #09 Trouble	41
OCU 1 Trouble	17		Remote Annunciator #10 Trouble	42
OCU 2 Trouble	18		Remote Annunciator #11 Trouble	43
OCU 3 Trouble	19		Remote Annunciator #12 Trouble	44
OCU 4 Trouble	20		Remote Annunciator #13 Trouble	45
OCU 5 Trouble	21		Remote Annunciator #14 Trouble	46
OCU 6 Trouble	22		Remote Annunciator #15 Trouble	47
OCU 7 Trouble	23		Remote Annunciator #16 Trouble	48
OCU 8 Trouble	24		Remote Annunciator #17 Trouble	49
OCU 1 Type Error	25		Remote Annunciator #18 Trouble	50

Continued on next page.

Default System/Panel Troubles continued:

System/Panel Trouble Type	Enumeration		System/Panel Trouble Type	Enumeration
Remote Annunciator #19 Trouble	51		Node #21 Network Failure	100
Remote Annunciator #20 Trouble	52		Configuration Data Upload	76
Remote Annunciator #21 Trouble	53		Operating Program Download	77
Remote Annunciator #22 Trouble	54		Event Log Data Upload	78
Remote Annunciator #23 Trouble	55		Maintenance List Data Upload	79
Remote Annunciator #24 Trouble	56		Node #01 Network Failure	80
Remote Annunciator #25 Trouble	57		Node #02 Network Failure	81
Remote Annunciator #26 Trouble	58		Node #03 Network Failure	82
Remote Annunciator #27 Trouble	59		Node #04 Network Failure	83
Remote Annunciator #28 Trouble	60		Node #05 Network Failure	84
Remote Annunciator #29 Trouble	61		Node #06 Network Failure	85
Remote Annunciator #30 Trouble	62		Node #07 Network Failure	86
CIM Trouble	63		Node #08 Network Failure	87
Loop 1 Power Fault	64		Node #09 Network Failure	88
Loop 2 Power Fault	65		Node #10 Network Failure	89
Loop 3 Power Fault	66		Node #11 Network Failure	90
Loop 4 Power Fault	67		Node #12 Network Failure	91
Trouble Input	68		Node #13 Network Failure	92
Drill	69		Node #14 Network Failure	93
Silence	70		Node #15 Network Failure	94
Network Ground Fault	71		Node #16 Network Failure	95
Network Failure Port A	72		Node #17 Network Failure	96
Network Failure Port B	73		Node #18 Network Failure	97
Duplicate Node Numbers	74		Node #19 Network Failure	98
Configuration Data Download	75		Node #20 Network Failure	99

Continued on next page.

Default System/Panel Troubles continued:

System/Panel Trouble Type	Enumeration		System/Panel Trouble Type	Enumeration
Node #22 Network Failure	101		Node #57 Network Failure	126
Node #23 Network Failure	102		Node #58 Network Failure	127
Node #24 Network Failure	103		Node #59 Network Failure	128
Node #25 Network Failure	104		Node #60 Network Failure	129
Node #26 Network Failure	105		Node #61 Network Failure	130
Node #27 Network Failure	106		Node #62 Network Failure	131
Node #28 Network Failure	107		Node #63 Network Failure	132
Node #29 Network Failure	108		Node #64 Network Failure	133
Node #30 Network Failure	109		Main Power Fault	134
Node #41 Network Failure	110		Trouble	135
Node #42 Network Failure	111			
Node #43 Network Failure	112			
Node #44 Network Failure	113			
Node #45 Network Failure	114			
Node #46 Network Failure	115			
Node #47 Network Failure	116			
Node #48 Network Failure	117			
Node #49 Network Failure	118			
Node #50 Network Failure	119			
Node #51 Network Failure	120			
Node #52 Network Failure	121			
Node #53 Network Failure	122			
Node #54 Network Failure	123			
Node #55 Network Failure	124			
Node #56 Network Failure	125			

Appendix 1.4. Unsupported Features

Any functionality not specifically specified as supported above can be regarded as excluded. In particular:

1) Panel configuration messages such as those detailed in sections 8 and 9 of the Integlex specification are excluded.
2) Legacy messages such as those directed at the LaserCOMPACT device are excluded
3) Zone status' are excluded since there are no direct messages coming from the panel that indicate the status of the specific zones.

Appendix 1.5. Zone Data

The driver does not process zone data reported in event messages.

Appendix 1.6. Simulation Data

The Integlex Panel can be put in simulation mode and in this mode it reports events but marks them as simulated events. The driver does not differentiate between actual and simulated events.

Appendix 1.7. Driver Error Messages

Error Message We have shown place holders for the parts of the message which change. %s is a place holder for a text string. %d is a place holder for a number %c is a place holder for an alpha character.	Explanation and corrective action ***FYI messages are informational and do not require a corrective action. Simply use them to confirm configuration / behaviors are what you expect.***
IGLX#01 Err Dont recognize <%s>	This is a configuration error. It means that the value specified in the configuration for the parameter known as Integlex_Function is not recognized. See section 4.4.2 Driver Related Map Descriptor Parameters Correct the config, download and restart the Gateway for changes to take effect.
IGLX#02 Err Dont recognize <%s>	This is a configuration error. It means that the value specified in the configuration for the parameter known as Integlex_Function is not recognized. See section 4.4.2 Driver Related Map Descriptor Parameters Correct the config, download and restart the Gateway for changes to take effect.
GLX#03 Err Cant open <%s>	You should never see this message. If you do please capture a log using the Diagnostic

	tool and send it for analysis when you report the error.
IGLX#04 Err File <%s> doesnt have line=%d	You should never see this message. If you do please capture a log using the Diagnostic tool and send it for analysis when you report the error.
IGLX#05 Err File <%s> line=%d does not have 4 tokens	You should never see this message. If you do please capture a log using the Diagnostic tool and send it for analysis when you report the error.
IGLX#06 Err. Rcvd Line. Does not begin STX 0x%02x %c	If this is an occasional error you could ignore it but even one lost message can be important since this driver cannot poll for an update. This error is printed when the oncoming byte stream is corrupted. This could be because of noise (occasional) or because the connection settings are wrong. In trying to diagnose this problem we suggest you capture a log using the Diagnostic tool and send it when you report the error.
IGLX#07 Err. Rcvd Line bad format	If this is an occasional error you could ignore it but even one lost message can be important since this driver cannot poll for an update. This error is printed when the oncoming byte stream is corrupted. This could be

	because of noise (occasional) or because the connection settings are wrong. In trying to diagnose this problem we suggest you capture a log using the Diagnostic tool and send it when you report the error.
IGLX#08 FYI System Trouble Strings loaded from file=systrb.ini	If systrb.ini is found and loaded this message is printed.
IGLX?09FYI AddSysTrb=%03d <%s>	If systrb.ini is found and loaded then this message is printed for each line that is loaded from the file.
IGLX#10 Err No space to add SysTrb Max=%d (%d)	Systrb.ini has more lines of system troubles than the driver can accommodate. Review your configuration.
IGLX#11 Err. Event ignored. Unknown Node=<%s>	It is critical that Node Names match the node names allocated in the Integlex configuration of the FACP's. There should be one node in the config file for each node you wish to store data from. If an event is reported from a panel that has not been configured (ie whose number / name is not in the config) then this message will be printed.
IGLX#12 FYI Classifications loaded from file=classi.ini	If classi.ini is found and loaded this message is printed.
IGLX#13 FYI AddClassi=%03d <%s>	If classi.ini is found and loaded then this message is printed for each line that is loaded from the file.
GLX#14 Err No space to add classi.ini Max=%d (%d)	classi.ini has more lines of system troubles than the driver can accommodate. Review your configuration.

IGLX#15 Err. Classification=<%s> not known	The classification of a device that reported an event has not been recognized. You may need to add the new classifictation to the the list of known classifications. To do this you need to make a file with all the classifications including the new one and download the file to the fieldserver. On the next reboot it will be loaded.
IGLX#16 Err. Cant find any MD's to store data.	The driver was attempting to store data for an event associated with a device but could not find a Map Descriptor which defines where the particular event should be stored. You may have chosen to ignore this event but doing a configuration which does not include it OR you may need to update your configuration, adding Map Descriptors for the event type, the node and/or the loop.
IGLX#17 Err. Mismatch Nodes %d %d	An internal error occurred when storing events. Please send a copy of your configuration when reporting this error to support.
IGLX#18 FYI %s DA=%s Off=%d St=%d	Each time a device event is stored the driver prints this line. It tells you the Data Array, offset and the value to be set in the DA.
IGLX#19 Err. Cant find DA=%s to store Summary	The Map Descriptor indicates that summary data must be stored. When the driver looked for the specified Data Array it could not find it. This is a configuration error.
IGLX#20 FYI DA=%s Off=%d val=%%x%02x Bit=%d Before%s	Summary data for an event is being stored. The driver informs you what the Data Array, offset, bit number and the value found at the specified offset in the Data Array, before the summary was updated.

IGLX#21 FYI DA=%s Off=%d val=%%x%02x After%s	As above, but after the update.
IGLX#22 Err. Cant find any place to store - %s. IGLX#22 Err. Node=%d Dev=%d Loop=%d Zone=%d	The driver want to store and event but cant find a Map Descriptor which tells it where to store the data. This may be ok – you may have chosen to configure only some events, some loops, some panels.
IGLX#23 Err. Node=%d Cant find DA to store SysTroubles	The driver cant find the Data Array specified to store the system trouble.
IGLX#24 FYI DA=%s Node=%d Off=%d St=%d Trb=%s	The driver is storing a system trouble and reporting to you where it was stored. The Data Array and offset, Node, whether it is an On/off event and the event type are reported.
IGLX#25 Err. Cant recognize System Trouble String IGLX#25 Err. string=<%s>	The driver want to store a system Trouble event but cant find a Map Descriptor which tells it where to store the data. This may be ok – you may have chosen to configure only some events, some loops, some panels.
IGLX#26 Err. Cant find any MD's to store data. IGLX#27 Err. Mismatch Nodes %d %d	The driver cant find any Map Descriptors to to store the system trouble for the specified node.
IGLX#28 Err. Node=%d SysTrouble No Storage.	An internal error. Please capture a diagnostic using the FieldServer utilities and send it to support for analysis.
IGLX#29 FYI Reset time start=%ld	When a System reset occurs the driver prints this message
IGLX#30 Err. Cant find any MD's to store data.	In doing a system reset, the driver could not find any Map Descriptors which require resetting. This is not illegal but unusual. Please review your configuration or send it for review by our suppor team.
IGLX#31 FYI. Classifications retained after	When a system reset is performed the

reset. IGLX#31 FYI. MD=%s DA=%s	classification data is not cleared. This message confirms this and tells you the MD and Data Array that were not cleared during the reset.
IGLX#32 Err. Node=%d System Reset - No DA's.	In doing a system reset, the driver could not find any Data Arrays for the specified node which require resetting.
IGLX#33 FYI Reset time end=%ld delta=%ld	When a System reset occurs the driver prints this message when it is complete.
IGLX#34 Err. No place to store SysTroubles IGLX#34 Err. string=<%s>	This could be normal if you have chosen not to store system/panel troubles from all the panels. Review your configuration and ensure that Map Descriptors are defined to capture trouble messages from all panels and that they Data Arrays referenced by the Map Descriptors exist.
IGLX#35 FYI DA=%s Node=%d Off=%d St=%d Trb=%s	A system Trouble was stored. The driver reports key information.
IGLX#36 Err. Cant recognize System Trouble String IGLX#36 Err. string=<%s>	The driver can't store a system trouble because it does not recognize the system trouble description string. It may be necessary to add the new trouble string to systrb.ini and to download the file to the Gateway.
IGLX#37 Err. Cant find DA=%s to store Summary	The Map Descriptor indicates that Full summary data must be stored. When the driver looked for the specified Data Array it could not find it. This is a configuration error.
IGLX#38 FYI DA=%s Off=%d val=%%x%02x Bit=%d FBefore%s	Full Summary data for an event is being stored. The driver informs you what the Data Array, offset, bit number and the value found at the specified offset in the Data

	Array, before the summary was updated.
IGLX#39 FYI DA=%s Off=%d val=%%x%02x FAfter%s	As above, but after the update.
IGLX#40 Err. The Function Name must be specified.	Configuration Error. The parameter Integlex_Function has been omitted. It cannot be blank. Correct the error in the config. Download the corrected file. Reset the gateway for changes to take effect.
IGLX#41 Err. The Summary DA must be specified.	Configuration Error. The parameter 'Integlex_Summary_DA' cannot be left blank. If you don't want to specify a Data Array then use a dash Correct the error in the config. Download the corrected file. Reset the gateway for changes to take effect.
IGLX#42 Err. The Function Name must be specified.	Configuration Error. The parameter 'Integlex_Full_Summary_DA' cannot be left blank. If you don't want to specify a Data Array then use a dash. Correct the error in the config. Download the corrected file. Reset the gateway for changes to take effect.

Appendix 1.8. Exposing Driver Stats

The driver makes some of its operating statistics available in a Data Array where they can be read by a remote client. The lines from the example below can be cut and pasted into a configuration file.

```
Data_Arrays,
Data_Array_Name,        Data_Format,        Data_Array_length,
integlex_stats,         UINT32,             1000,
```

Offset	Description
0	Increments each time the driver receives a complete message from the panel
1	Increments each time a message is received that the driver does not have the capability of processing.
2	Increments each time a system trouble events is processed and the driver does not recognize the system trouble string (event). It may require that you update the system trouble file.
3	Increments each time a 'System Initialized' message is received.
4	Increments each time a 'System Normal' message is received.
5	Increments each time a 'System Trouble' message is received.
6	Increments each time a 'Representative Trouble' message is received.
7	Increments each time a 'Drill' message is received.
8	Increments each time a 'Silence' message is received.

Offset	Description
9	Increments each time a 'System Reset' message is received.
10	Increments each time a 'Module Disable' message is received.
11	Increments each time a 'NAC Disable' message is received.
12	Increments each time a 'CMS Disable' message is received.
13	Increments each time a 'Output Disable' message is received.
14	Increments each time a 'Alert' message is received.
15	Increments each time a 'Action' message is received.
16	Increments each time a 'Alarm' message is received.
17	Increments each time a 'Alarm F1' message is received.
18	Increments each time a 'Alarm F2' message is received.
19	Increments each time a 'Confirm' message is received.
20	Increments each time a 'Verify' message is received.
21	Increments each time a 'Supervisory' message is received.
22	Increments each time a 'No Answer' message is received.
23	Increments each time a 'Comm Fault' message is received.
24	Increments each time a 'Disable' message is received.
25	Increments each time a 'Type Error' message is received.
26	Increments each time a Fault' message is received.
27	Increments each time a 'Data Error' message is received.
30	Increments System Trouble String was not recognized.
31	Increments each time a System Trouble report cannot be stored – typically a consequence of the message being ignored because system troubles for the node are not configured.

Offset	Description
32	Increments each time a device event message could not be stored because no storage was defined for the Node, Loop or the event type
33	Increments If the driver attempts to store summary data but the Data array could not be found.
34	Increments when there is no MD to store an event.
35	Increments when there are no MD's to store a system trouble.
36	Increments each time the classification was not recognized.
37	Debug Mode. Set the value of this location to 1 to have the driver dump additional debug info into the log.
38	Like #33 but for a full summary.
39	Increments each time a message is received which reports a point as Active
40	Increments each time a message is received which reports a point as Inactive

Appendix 1.9. Revision History

Date	Resp	Format	Driver Ver.	Doc. Rev.	Comment
2012Aug07	PMC		1.0	1.0	Created.
2012Aug23	PMC		1.0	2.0	Updated
2012Sep07	PMC		1.0	3.0	Updated
2012Sep12	GFM		1.0	4.0	QC