Gamewell

Description

All Gamewell Fire Alarm panels are equipped with a serial port which produces panel, circuit or device status messages. This driver is designed to process these messages and store this status information in numeric form. The numeric value will indicate the type of event being reported and the storage location in the FieldServer's data arrays is (configurable &) dependent on the origin of the message (panel / circuit / device). Additional information such as event date and time and descriptions are ignored.

The driver is capable of supporting a panel configured to supervise the port by responding to the panel's supervision queries.

This is a passive client driver. The driver listens passively for unsolicited messages produced by the Gamewell panel.

Design Basis: Gamewell serial port protocol specification "IF 600r7 Message Stream" (not dated) and "SmartNet Data stream information" (not dated).

The driver is capable of exposing communication statistics in a FieldServer Data Array so they can be monitored by a remote device.

Formal Driver Type

Serial, Passive Client

Compatibility

FieldServer Model	Compatible
ProtoCessor	No
ProtoCarrier	No
ProtoNode	No
ProtoAir	No

FieldServer Model	Compatible
QuickServer FS-QS-10xx	No
QuickServer FS-QS-12xx	Yes
QuickServer FS-QS-20xx	Yes
QuickServer FS-QS-22xx	Yes
QuickServer FS-QS-3x10-F	Yes

Connection Information

Connection Type: RS-232 or RS-485 (w/converter)

Baud Rates: Gamewell Panel: 2400; Driver: 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 28800, 38400, 57600, 115200

Data Bits: Panel: 8; Driver: 7, 8

Stop Bits: Panel: 1; Driver: 1, 2

Parity: Panel: None; Driver: Odd, Even, None

Multidrop Capability: No

Devices Tested

Device	Tested (Factory, Site)
IF610	@MSA Safety, Milpitas, May 2002. Tested port supervision, alarm generation, message filtering using data types, system reset clears data arrays. Test to send reset/back/silence from FieldServer passed.

Communication Functions

Listen – The driver listens passively for status messages, parses them looking for Node, Status, Circuit/Device and action information and stores data based on this information.

Write – The driver can send an Ack, Silence and Reset command. (To be provided in a later release.)

Supervision Query / Response

The driver clears its data arrays when the following messages are received.

Status:NORMAL 08/31/95 16:23

System Idle

Data Types Supported

Data_ Type	Type of Information Stored	Notes
Any	Stores Status Information	Status: ???. Stores non-zero value for any not-normal status.
Alarms	Stores Status Information	Status: ALARM sets array non-zero. NORMAL sets array to zero.
Faults	Stores Status Information	Status: FAULT sets array non-zero. NORMAL sets array to zero.
Events	Stores Status Information	Status: EVENT sets array non-zero. NORMAL sets array to zero.
Bus	Stores Status Information	Status: BUS sets array non-zero. NORMAL sets array to zero.
Comm	Stores Status Information	Status: COMM sets array non-zero. NORMAL sets array to zero.
Control	Stores Status Information	Status: CONTROL sets array non-zero. NORMAL sets array to zero.
Ack	Stores Status Information	Status: ACK sets array non-zero. NORMAL sets array to zero.
Signal Silence	Stores Status Information	Status: SIG SIL sets array non-zero. NORMAL sets array to zero.
Troubles	Stores Status Information	Status: FAULT sets array non-zero. NORMAL sets array to zero.
Supervisories	Stores Status Information	Status: EVENT and Action contains ‘Supv. Event in’ sets array non zero. Or Status: SUPV and any action. Normal sets array to zero.
Action_Numbers	Stores Action Information	Value based on contents of “Action” Field.
Action_Bits	Stores Action Information	Sets bit whose offset is based on contents of ‘Action’ Field.
Dump	Dump’s ignored messages for user review	

Driver Limitations and Unsupported Features

TThe driver stores a value representing the type of status message received. A table of status types vs. values is provided in the driver manual. Each message is inspected for circuit/device information. If none is present the message is assumed to report a status event for the panel. If the one or both are present then the circuit / device number is used to determine the storage location.

The driver can store a value to represent the status of a point (device / Circuit / panel) and/or a value to represent the ‘action’ that caused the most recent message to be sent.

For messages reporting a status event for a circuit / device the driver uses only the device number to determine the location to store the indicating value.

The driver does not maintain an event / alarm history.

The value zero will be used to represent normal.

The driver is programmed with a list of status types and action types that it recognizes. In the event that unrecognized information is found, the driver will store special value to indicate this. The driver provides a method which allows the user to extend the list of recognized status types and actions.