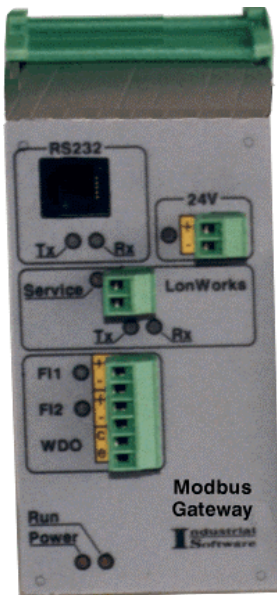


# Installation Instructions for LonWorks Modbus Gateway

Date last modified: 30-01-2001

## LonWorks Modbus Gateway



### GENERAL INFORMATION

The LonWorks Modbus Gateway (LMG) provides integrated control solution, providing a bridge between LonWorks network and Modbus devices. The benefits of gateway include:

- Programmable Comm parameters for Modbus side (baudrate, parity, address, ASCII/RTU)
- Configuration uses config network variables from LonWorks side
- Flexible routing tables
- Powerfull 16-bit microcontroller for Modbus side

### INSTALLATION

Gateways are shipped from factory in UNCONFIGURED state.

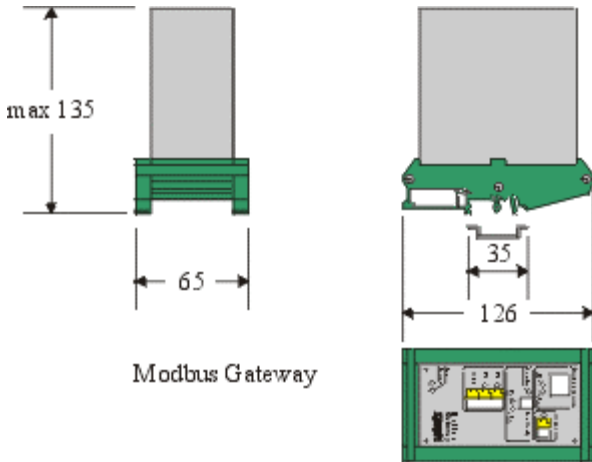
All devices have unique serial number (NEURON ID).

After physically installation of device you may use **Find and Wink** or **Neuron ID** installation scenario.

## INSTALLATION - MECHANICAL

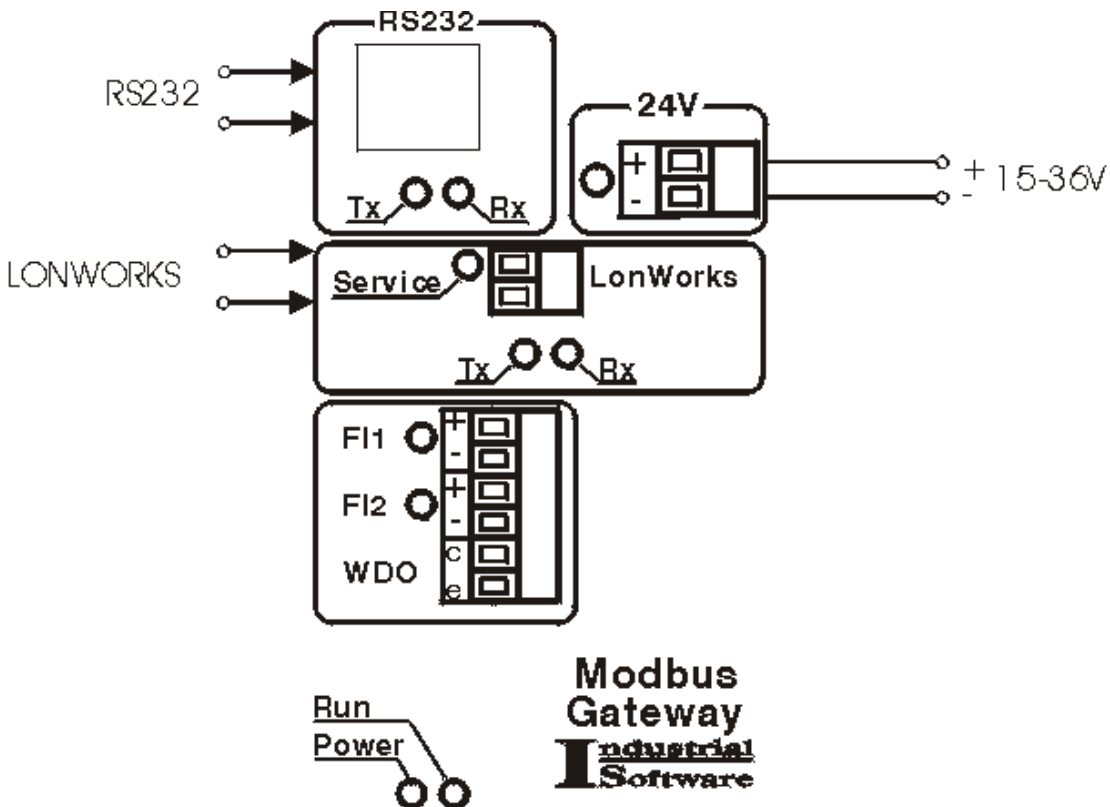
To mechanically install the Modbus Gateway, proceed as follows:

1. Drill two clearance holes for mounting DIN rail. Mount it.
2. Mount the module on the rail.



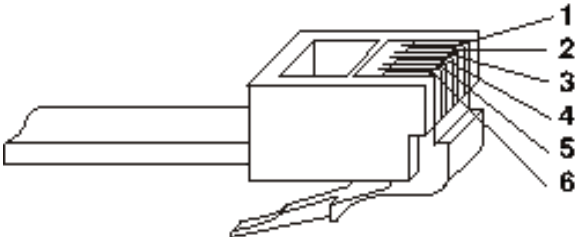
LonWorks Modbus Gateway

## INSTALLATION - ELECTRICAL





The following tables provide connector pinout information for each control module:

### Connector RS 232

Connections	PART OF ....	Descriptions
	1	<b>GND</b> 1
	2	<b>CTS</b> 1
	3	<b>RTS</b> 1
	4	<b>TxD</b> 1
	5	<b>RxD</b> 1
	6	<b>Vcc*</b> 1

5VDC - Used to power external Galvanically insulated RS232/RS485 converter if needed. 100 mA maximum sink.

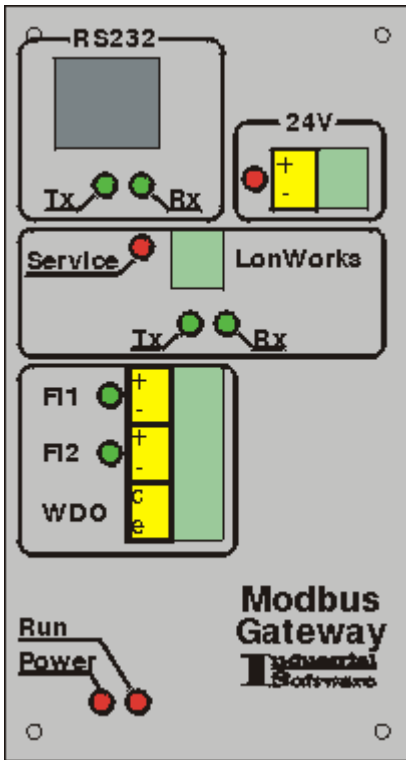
### Module Power Connection

Module Power Connection	Micro Pin	Function
	1	- Power Supply
	2	+ Power Supply
Module Network Connection	Micro Pin	Function
	1	FTT-10 or TP/XF-1250
	2	FTT-10 or TP/XF-1250

### SPECIFICATIONS

<b>Electrical</b>	
Module Power Voltage	15 - 36 VDC
Module Current Consumption	80 mA / 24VDC
Interface Capacity	48 read/write registers, 96 read/write coils
Galvanic insulation	LonWorks to Modbus, Power to Modbus, Power to LonWorks
LonWorks Voltage &Current	No need of power

<b>Network</b>		
Topology	Bus or free, single or double terminated.	
Media	Twisted pair.	
Interface	Transformer isolated polarity insensitive.	
<b>Environmental</b>		
Temperature		
	Operational(ambient)	-10 oC to +85 oC
	Storage	-40 oC to +85 oC
Humidity	95% RH,non-condensing	
Shock	10 G	
Vibration	2 G, at 10 to 500 Hz	
Electromagnetic compability	IEC801, level 3	
<b>Standards</b>		
IEC	IEC 801, IEC1131-1	
<b>Physical</b>		
Size( max. dimensions)	126mm*135mm*65mm	
Mounting	DIN Rail	
Weight	Enclosed without DIN Rail - apr. 250 g.	
Housing /Material	Poliamide PA-F fiber reinforced , aluminum Face - polycarbonat	
<b>Termination's</b>	Plug-in direction vertical to conductor axis, 3.81 mm step	
Recommended wire size	AWG28-16 (24-14) stranded or solid	
Communication Indicators	Modbus side Tx, Rx - green LonWorks side Tx, Rx - green	
Additional Indicators	Power - red, Service( LonWorks specific) red	
Front Panel	See Front Panel View bellow	



Front Panel View

### Principal of operation of Modbus/LonWorks Gateway

Modbus/LonWorks Gateway makes a route between its Modbus and LonWorks side.

When a value in some Modbus is changed from the value of previous poll, it's propagated over

LonWorks if the corresponding variable( as defined by mapping table) is connected.

When a LonWorks side input variable is written, its value is downloaded to it's corresponding

Register at Modbus side.

Note the restrictions, that Neuron Based nodes ( also Gateway ) have:

1. No more than 48 registers can be mapped to single gateway.
2. Each Gateway has only 15 address table entries, and each output variable connection uses one entry.

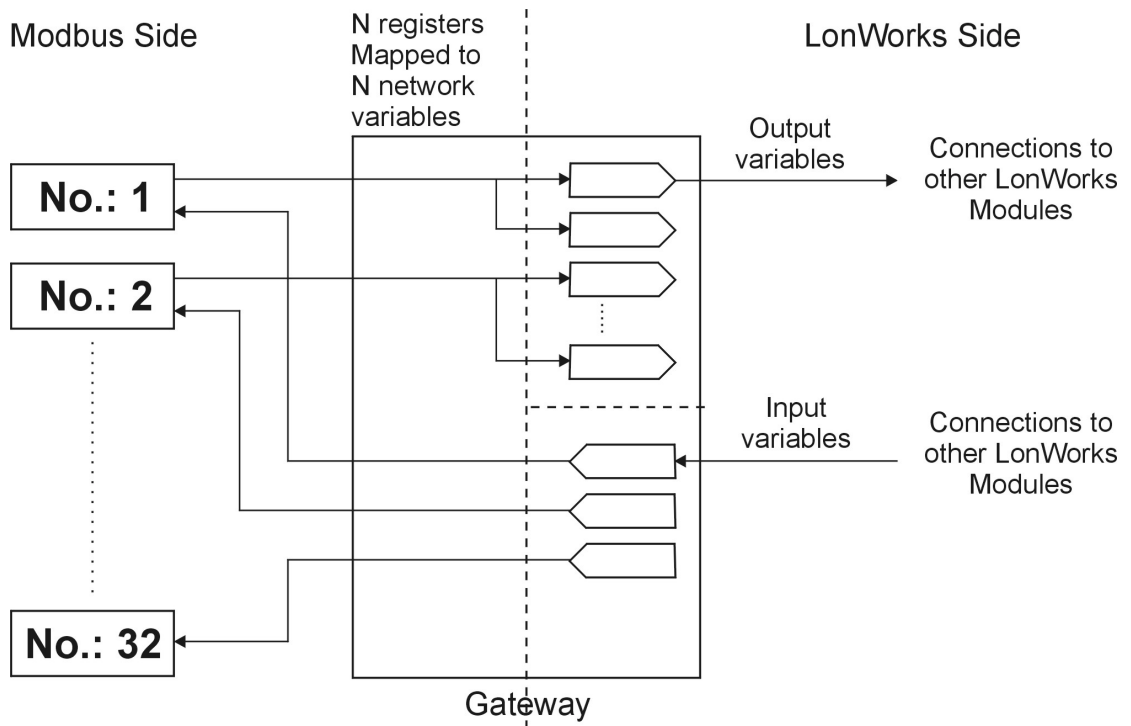


Fig. 1

## Gateway Structures, Types and Variables

////////// Definitions of New Types //////////

typedef struct

```

{
    unsigned uldx;
    unsigned uModNum;
    unsigned uFun;
    unsigned uAddrHi;
    unsigned uAddrLo;
    unsigned uNumPHi;
    unsigned uNumPLo;
    unsigned uNumByte;
    unsigned uNumVar;
}StTpAddr;           // 9 Bytes

```

```
typedef struct
{
    unsigned    uAct;
    StTpAddr    sTbl;
} StTpMapTbl;           // 10 Bytes
```

```
typedef struct
{
    unsigned    uNum;
    unsigned uBaud;
    unsigned uPar;
    unsigned uPollHi;
    unsigned uPollLo;
    unsigned uTimeHi;
    unsigned uTimeLo;
} StTpCnfg;           // 7 Bytes
```

```
typedef struct
{
    unsigned    uData[31];
} StTpTest;
```

// Rarallel\_io\_Interface Structors

```
typedef struct
{
    unsigned    uLen;
    unsigned    uCode;
    unsigned    uData[DATA_SIZE-2];
} parallel_io_interface;
```

typedef struct

```
{  
    unsigned    uLen; // 8 Bytes  
    unsigned    uCode;    // 1-Code  
    StTpCnfg    sCnfg;// 7 Bytes  
    }ST_PIO_Cnfg;
```

typedef struct

```
{  
    unsigned    uLen; // 11 Bytes  
    unsigned    uCode;    // 2,3-Code  
    unsigned    uAct; // 1 Byte  
    StTpAddr    sTbl;    // 9 Bytes  
    }ST_PIO_Tbl;
```

typedef struct

```
{  
    unsigned    uLen; // 11 Bytes  
    unsigned    uCode;    // 2,3-Code  
    StTpTest    sData;// 31 Bytes  
    }ST_PIO_Req;
```

typedef struct

```
{  
    unsigned    PioCount;  
    unsigned    uByte[6];  
    }StTpState;
```

## //////////////////// Network Variables //////////////////////

```
/* 00 */ network input StTpMapTbl          nviMapTbl;
/* 01 */ network input SNVT_count          nviModBusNum;
/* 02 */ network input SNVT_count          nviBaudRate;
/* 03 */ network input SNVT_count          nviParity;
/* 04 */ network input SNVT_count          nviPollTime;
/* 05 */ network input SNVT_count          nviMwaitS;
/* 06 */ network input SNVT_count          nviMaxSendTime;
/* 07 */ network input SNVT_lev_disc        nviModeGtw;
/* 08 */ fastaccess network output SNVT_state nvoModBusErr[2];
/* 10 */ fastaccess network input SNVT_count nviLonToMod[NVI_NUM];
/* 34 */ fastaccess network output SNVT_count nvoModToLon[NVO_NUM];
/* 58 */ network input StTpTest            nviRegModData;
/* 59 */ network output StTpTest           nvoRespModData;
/* 08 */ network input SNVT_state          nviModBusSts;
far network input StTpState                nviState;
```

## Network Variables Description

### **nviMapTbl**

This variable is used by configuration tool to download and upload configuration tables.

### **nviModBusNum**

This variable is used to specify MODBUS Gateway's Number in MODBUS network, when it works as Modbus Slave. Not used when Gateway is Modbus Master.

### **nviBaudrate**

Modbus side Baudrate. Valid Values – 1200, 2400 , 4800 , 9600, 19200, 38400. This values represent corresponding baudrates in bits/second.

Default value: 19200

**Caution: Write to this network variable initiates writing in EEPROM, so frequently writing may cause an EEPROM damage !**

## nviParity

Modbus side Parity configuration. Valid values – 0 for No Parity, 1 for even parity, 2 for odd parity.

**Caution: Write to this network variable initiates writing in EEPROM, so frequently writing may cause an EEPROM damage !**

## nviPollTime

Modbus Polling interval in milliseconds – Time between 2 polls of Modbus devices, if “Table mode” is used. ( see nviModeGtw ). Valid Values – 40 to 20000 ms.

**Caution: Write to this network variable initiates writing in EEPROM, so frequently writing may cause an EEPROM damage !**

## nviMwaitS

Modbus Side Timeout in milliseconds – If Modbus Gateway tries to read from or write to some device, and don't receives query in this interval, it generates an error for this device and sets to 1 appropriate bit in nvoModBusErr[2]; Be Careful when configuring this timeout – at slower baudrates it have to be higher. **It also have to be lower then nviPollTime.**

## nviMaxSendTime

Time in miliseconds, between two propagates of network variables, if their values are not changed.

Valid values – 0 – never propagate if there is no change ; 500 – 30000 .

## nviModeGtw

Mode of functioning of Gateway. Valid Values:

ST\_OFF (0) – **“Query” mode**

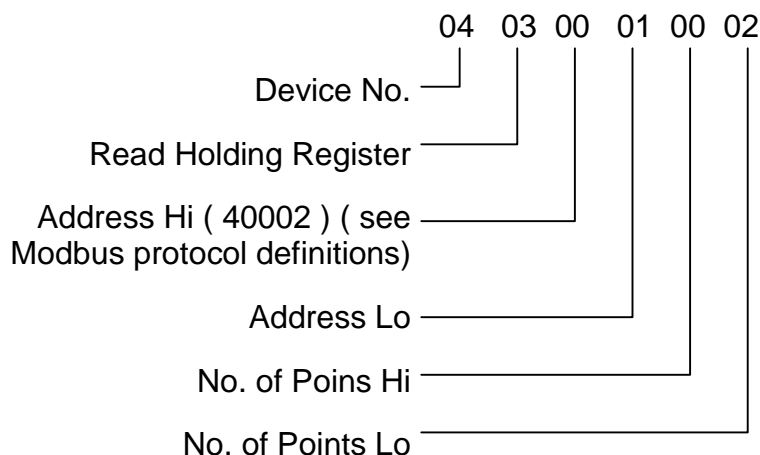
In this mode Gateway polls Modbus device only if nviRegModData written with appropriate Modbus Message and returns result in nvoRespModData.

**Gateway doesn't use any internal tables in this mode !**

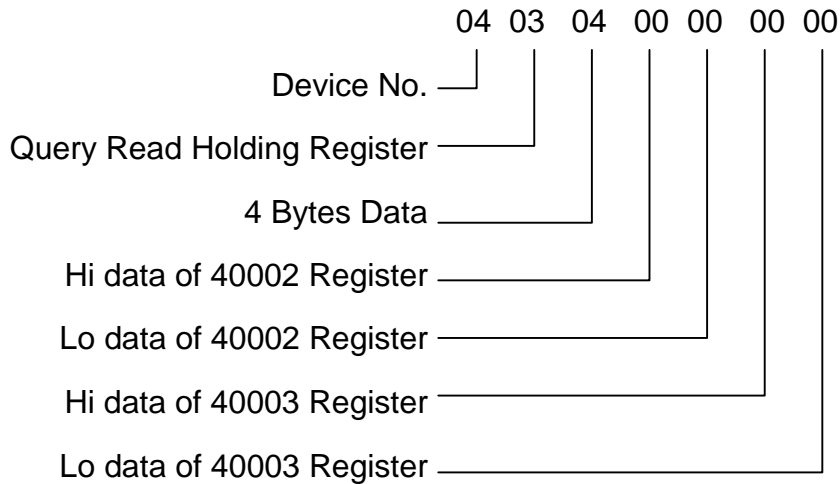
**Gateway forms CRC by itself !**

Example: ( all values in the example are hexadecimal )

write to nviRegModData :



Gateway returns in nvoRespModData :



### ST\_ON (4) – **“Table” mode**

In this mode Gateway uses tables , downloaded by configuration tool to poll devices sequentially and send information to nvoModToLon[NVO\_NUM].

If some value is written to nviLonToMod[NVI\_NUM] , this value is downloaded to Modbus device, as this is described with configuration tool.

### **nvoModBusErr[2]**

Each bit in these variables represents the state of one Modbus device .

0 – O.K

1 – Modbus device doesn't answers in nviMwaitS time after poll.

LSB( bit 0 ) of nvoModBusErr[0] is for device 1

MSB(bit 1 ) of nvoModBusErr[0] is for device 16

LSB( bit 0 ) of nvoModBusErr[1] is for device 17

MSB(bit 1 ) of nvoModBusErr[1] is for device 32

### **nviModBusSts**

Message level Errors at Modbus side