P-NET to RS232

PD 3940

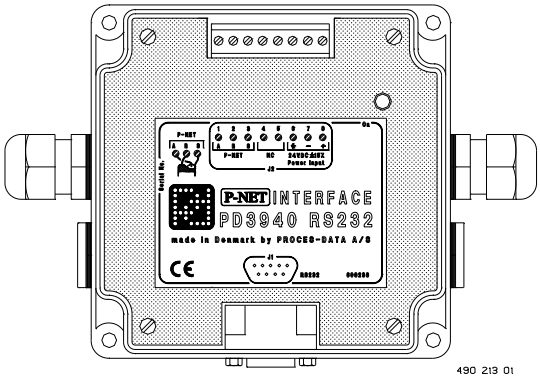
Manual

GB *January 1997*

PROCES-DATA A/S, Navervej 8, 8600 Silkeborg, Denmark, Phone +45-87 200 300, fax + 45-87 200 301

Contents

1 General information.



The PD 3940 is a communication module, providing an interface from the P-NET fieldbus to RS232 devices, such as printers, card-readers, barcode readers, and other equipment not having built-in fieldbus facilities for P-NET. This enables such devices to be integrated into a P-NET fieldbus system. It can be used in conjunction with the varied collection of distributed P-NET input/output and control modules, which provide digital, analog, flow, weighing and power monitoring facilities.

Application examples:

- Printer interface, for printout of tickets, alarms and reports.
- Card-reader interface, for entrance control.
- Bar-code reader interface, for production control to track the flow of materials.

Configuration of the module for the functions required, and communication between the module and a control computer, is carried out via the P-NET. The PD 3940 module is designed to work in a multi-master environment, where it can be completely controlled via P-NET.

1.1 Features.

- Integrates RS232 devices with P-NET
- Fully configurable via P-NET.
- Baud rates from 300 to 76,800.
- Standard RS232 handshake signals.
- Galvanically isolated com. ports.
- Full duplex communication.
- Dynamic buffer size.
- P-NET Fieldbus Communication, European standard EN 50170, Vol. 1.
- RS232 port (9 pin male).
- IP53 mounting box.
- Panel mount facilities.
- EMC approved (89/336/EEC)

1.2 System description.

The PD 3940 P-NET to RS232 communication module provides an interface between the P-NET fieldbus and any RS232 device. It is connected to P-NET in the same way as any other node, and has a built in RS232 9-pin male connector. The PD 3940 is mounted in an IP53 polycarbonate mounting box. Four PG11 cable glands are provided, for connection of the P-NET fieldbus and power supply cables.

The PD 3940 is a P-NET slave, which has 2 channels, consisting of the Service channel (channel 0) and the Communication channel (channel 1).

The Service channel, which is mandatory in all P-NET modules, holds amongst other things, general information about the module, such as ID, serial no., P-NET node address and error status of the complete module. The Communication channel holds all data applicable to the RS232 port set up (baud rate, parity, handshake etc), input/output databuffers, and error the status of the channel. Errors reported include configuration errors, communication errors and module errors.

It is possible to set a Simulation enable bit, for test purposes. If Simulation is selected, direct access to the input buffer is allowed, and data from the RS232 port will be ignored.

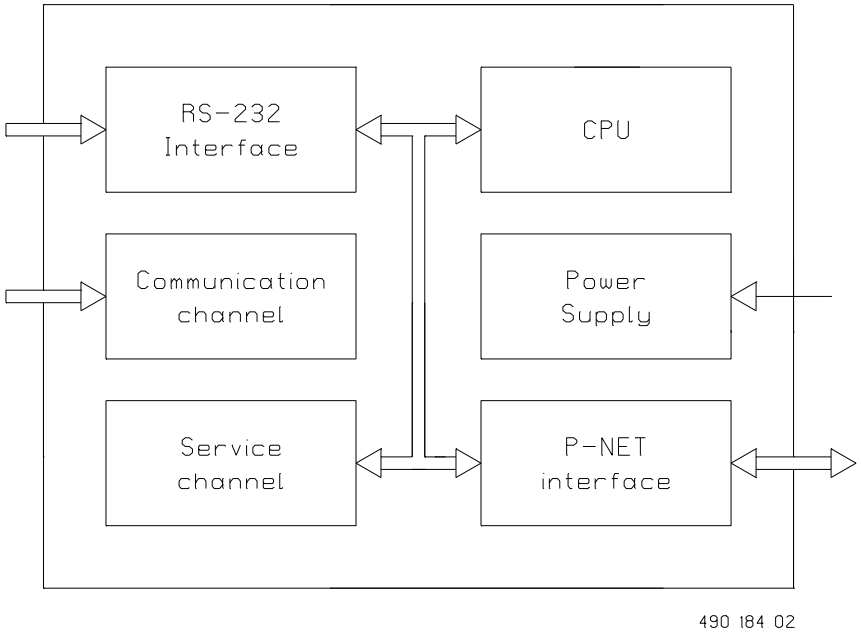
The Reservation facility allows a master to reserve the module for a certain period of time. In a multi-master network, this feature can prevent other masters from accidentally writing data to the output buffer while, for example, printing a report.

If the Dynamic buffer enable bit is selected, a read access to the input buffer will return the actual number of bytes, which have been received at the time of the read operation. This feature may, for example, be useful when the packages of data received on the RS232 port are not of a fixed length.

Data that are to be sent (e.g. to a printer) on RS232, must be stored in the Output buffer in the communication channel. Data received (e.g. from a barcode reader) are stored in the Input buffer, from where it can be read via P-NET.

1.3 Channel/registers

The PD 3940 module contains:



1 Service channel	(channel 0)
1 Communication channel	(channel 1)

A set of 16 variables, numbered from 0 - $F, is associated with each channel. To address a variable within a particular channel, a logical address called a SoftWire Number (SWNo), is used. The SWNo is calculated as: (channel number ∗ $10) + variable number within the channel.

Example: Variable 9 on channel 1 needs to be addressed.
The SWNo will therefore be $19.

Throughout the manual, the variables are depicted within tables. The variable names are standard identifiers, as defined in Process-Pascal.

1.4 Memory types

The PD 3940 stores data in different types of memory, depending on the value of a control variable following a reset or a power failure, and the state of write protection.

Some variables are stored in both non-volatile memory and volatile memory. The state of the module's WriteEnable register determines whether the contents are changed in both types of memory, or only in the volatile type.

The following memory types are listed in the channel definition tables.

Read Only

PROM ReadOnly
: The PROM is always write protected and can never be changed.

RAM ReadOnly
: The variable is stored in RAM and is only accessible for Reading.

Read, Protected Write

EEPROM RPW (Read, Protected Write)
: The EEPROM is always write protected directly following a reset. By setting WriteEnable to TRUE, the contents of the EEPROM can be changed. The contents of the EEPROM will remain unchanged during and after a power failure.

Read Write

RAM ReadWrite
: The variable can be changed instantly. After reset or a power failure, it's value is set to zero.

Read Write, Protected BackUp Write

RAM InitEEPROM
: The variable is stored in both RAM and EEPROM. After a reset, the variable is copied from EEPROM into RAM. When the variable is changed via P-NET, the value is changed in RAM. If WriteEnable is TRUE, the value is changed in both RAM and EEPROM when the variable is changed via P-NET.

2 Service channel (channel 0)

PD 3940 has a service channel containing variables and functions common to the entire module.

Variables on Service channel (channel 0). Channel identifier: **Service**

SWNo	Identifier	Memory type	Read out	Type
0	NumberOfSWNo	PROM Read Only		Integer
1	DeviceID	PROM Read Only	- - - - - - - -	Record
2				
3	Reset	RAM Read Write	Hex	Byte
4	PnetSerialNo	Special function	- - - - - - - -	Record
5				
6				
7	FreeRunTimer	RAM Read Only	Decimal	LongInteger
8				
9	ModuleConfig	EEPROM RPW	- - - - - - - -	Record
A				
B				
C				
D	WriteEnable	RAM Read Write	Binary	Boolean
E	ChType	PROM Read Only	- - - - - - - -	Record
F	CommonError	RAM Read Write	- - - - - - - -	Record

SWNo 0: NumberOfSWNo
This variable holds the highest SWNo in the module.

SWNo 1: DeviceID
The purpose of this record is to be able to identify the device. The record includes a registered manufacturer number, the type number of the module and a string, identifying the manufacturer.

The record has the following structure:

```
Record
    DeviceNumber    : Word;          (* Offset = 0 *)
    ProgramVersion  : Word;          (* Offset = 2 *)
    ManufacturerNo  : Word;          (* Offset = 4 *)
    Manufacturer    : String[20];    (* Offset = 6 *)
end
```

An example of the field values in the DeviceID record is shown below:

DeviceNumber = 3940
ProgramVersion = 101
ManufacturerNo = 1
Manufacturer = Proces-Data DK

SWNo 3: Reset
By writing $FF to SWNo 3, the module performs a reset, and ExternalReset in CommonError SWNo $F is set TRUE.

SWNo 4: PnetSerialNo
This variable is a record having the following structure:

```
Record
    PnetNo      : Byte;          (* Offset = 0 *)
    SerialNo    : String[20];    (* Offset = 2 *)
end
```

The serial number is used for service purposes and acts as a 'key' for setting the module's P-NET Node Address.

A special function is included for identifying a module connected to a network containing many other modules, having the same or unknown node addresses, and to enable a change of the node address via the P-NET.

Setting a new node address via P-NET, is performed by writing the required node address, together with the serial number of the module in question, into the PnetSerialNo at node address $7E (calling all modules). All modules on the P-NET will receive the message, but only the module with the transmitted serial number will store the P-NET node address.
An attempt to write data to node address $7E will give no reply. Consequently the calling master must disable the generation of a transmission error when addressing this node.

In the module, the SerialNo = "XXXXXXXXPD", is set by **PROCES-DATA**, and cannot be changed. The eight X`s indicate the serialnumber, and PD is the initials of PROCES-DATA.

SWNo 7: FreeRunTimer
FreeRunTimer is a timer, to which internal events are synchronised. The timer is of type LongInteger having a resolution of 1/256 Second.

SWNo 9: ModuleConfig
This variable is a record having the following structure:

```
Record
    Enablebit   : Bit8;          (* Offset = 0 *)
    Functions   : BYTE;          (* Offset = 1 *)
    Ref_A       : BYTE;          (* Offset = 2 *)
    Ref_B       : BYTE;          (* Offset = 3 *)
end
```

There are no applicable function settings for ModuleConfig in this module.

SWNo $D: WriteEnable
Write protected variables can only be changed (via P-NET) when WriteEnable is TRUE. Following a reset, WriteEnable is set to FALSE.
After modifying the contents of module EEPROM, WriteEnable should be set FALSE. An EEPROM sum check is calculated each time WriteEnable is changed from "TRUE" to "FALSE". The sum check calculation period is approximately 1 second. Consequently, the module should not be reset during this period, otherwise an EEPROM error can occur (see SWNo F: CommonError).

NB: Writing to EEPROM is limited to 10,000 cycles for each byte, including the sum check bytes.

SWNo $E: ChType
Each channel in an interface module is described in an individual ChType variable. This is a Record, consisting of a unique number for the channel type and a TRUE boolean value for each of the registers which are represented within a channel. The register number in a channel, corresponds to the index number in the boolean array. In addition to these fields, various other fields can be found in the record, which depends on the channel type.

```
Record
    ChannelType     : WORD;      (* Offset = 0 *)
    Exist           : Bit16;     (* Offset = 2 *)
    Functions       : Bit16;     (* Offset = 4 *)
end
```

For the Service channel, ChType has the following value:

ChannelType = 1

Exist =

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	0	1	0	1	0	0	1	1	0	1	1

Functions =

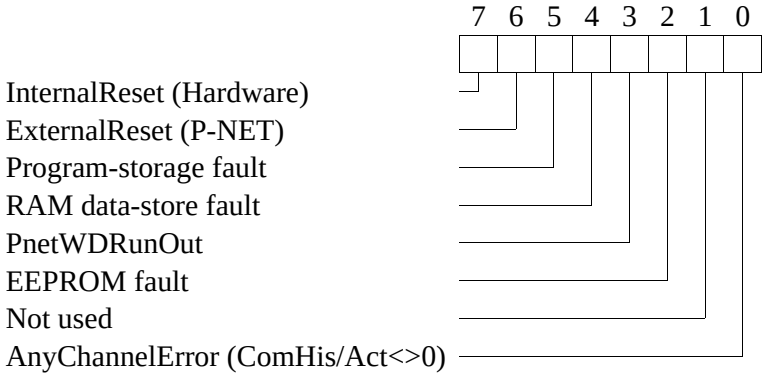
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SWNo $F: CommonError

The CommonError variable holds error information about all Channels. This variable is a record having the following structure:

```
Record
    ChError:
        Record
            His:Array[0..7] of Boolean;     (* Offset = 0 *)
            Act:Array[0..7] of Boolean;     (* Offset = 2 *)
        End;
    ComHis16:Array [0..15] of Boolean; (* Offset = 4 *)
    ComAct16:Array [0..15] of Boolean; (* Offset = 6 *)
End
```

The 8 bits in ChError.His and ChError.Act have the following meaning:



Bit 7 InternalReset is set TRUE if a reset is caused by a power failure, or if the power has been disconnected.

Bit 6 ExternalReset is set TRUE if a reset is caused by writing $FF to SWNo 3, Reset, via P-NET.

Bit 5 Program-storage fault is set TRUE if the self test finds an error in the program memory (PROM).

Bit 4 RAM data-store fault is set TRUE if the self test finds an error in the data memory (RAM).

Bit 2 EEPROM fault is set to TRUE if the self test finds an error in the data memory (EEPROM). The error may be corrected by setting and resetting WriteEnable.

Bit 0 AnyChannelError = 1 means that an error or an unknowledged error exists, in one or more channels.

The following function of ChError.His and ChError.Act is analogous for all Channels:

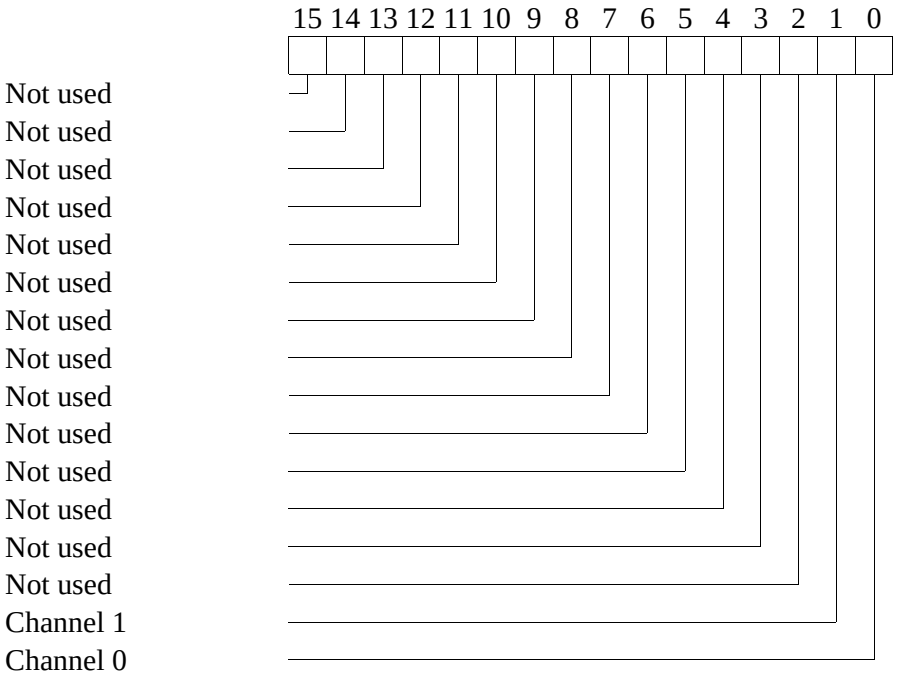
1 When an error occurs the corresponding bits in ChError.Act and ChError.His is set.
2 When the error disappears the corresponding bit is reset in ChError.Act.
3 After reading ChError.His, ChError.Act is copied to ChError.His.
4 Transmission responses from a module will include the Actual Data Error bit (DataError) set TRUE if ChError.Act <> 0.
5 The Historical Data Error bit (GeneralError) will be set TRUE in all responses from the module if ChError.His <> 0.

ComHis and ComAct are unique fields in the service channel, and hold an error status relating to all channels, where the bit number corresponds to the channel number. Each Channel has an error register, ChError. If ChError.His in a particular channel is<>0, the corresponding bit is set in ComHis. If ChError.Act in a particular channel is<>0, the corresponding bit is set in ComAct in the service channel. If the error disappears (ChError. Act = 0), the corresponding bit in ComAct is automatically cleared.

If the channels become error free, individual bits in ComHis will be cleared when reading ChError in each of the channels.

ComHis:=0 performs a special function, equivalent to reading all ChErrors.His in all channels.

ComHis / ComAct:



3 Communication channel (channel 1)

Variables on Communication channel Channel identifier: **Port_1**

SWNo	**Identifier**	**Memory type**	**Read out**	**Type**
10	OutputBuffer	RAM Read Write	- - - - - - - -	Buffer
11	InputBuffer	RAM Read Write	- - - - - - - -	Buffer
12	HandShake	RAM Read Only	Binary	Bit8
13				
14				
15				
16	Baudrate	EEPROM RPW	Decimal	LongInteger
17				
18	ReserveTimer	RAM Read Write	Decimal	Timer
19	ChConfig	EEPROM RPW	- - - - - - - -	Record
1A	ReservePreset	EEPROM RPW	Decimal	Real
1B	Reservation	RAM Read Write	Decimal	Boolean
1C	DatamodeParam	EEPROM RPW	- - - - - - - -	Record
1D	Maintenance	EEPROM RPW	- - - - - - - -	Record
1E	ChType	PROM ReadOnly	- - - - - - - -	Record
1F	CHError	RAM ReadOnly	Binary	Record

SWNo $10: OutputBuffer
If the port is configured for datamode out (ChConfig.Functions = DatamodeOut or Datamode-InOut), data in this buffer will be sent directly to the port.

The buffer is of the following form:

Buffer[10] of STRING[255]

Data is transmitted when there is at least one element in the buffer. Only the number of bytes corresponding to the actual length of the string, are sent.

When writing to OutputBuffer, the ReserveTimer (SWNo $18) is preset with the value found in ReservePreset (SWNo $1A).

SWNo $11: InputBuffer
If the port is configured for datamode in (ChConfig.Functions = DatamodeIn or DatamodeInOut), data received at the port is transferred directly to this buffer. The buffer has the following form:

Buffer[10] of STRING[255]

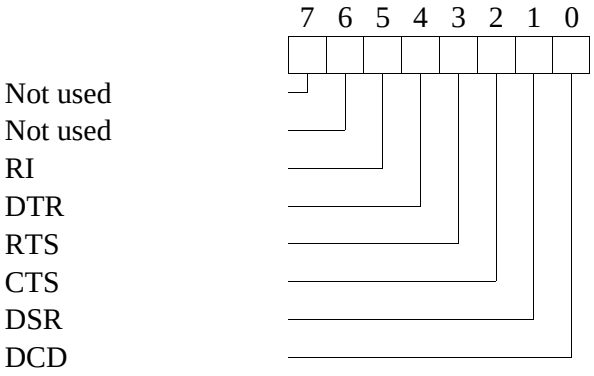
Data is transferred to the buffer elements depending on the conditions defined in DatamodeParam.

Since the electrical standard of the port is RS232, the standard rules for RS232 handshake signals apply. That is, CTS is activated whenever the port is ready to receive data in DatamodeIn, and deactivated if it is not ready (i.e. if not configured for DatamodeIn or the InputBuffer is full).

SWNo $12: HandShake

The HandShake register is a status register, showing the actual status of the signals on the RS232 terminals. A TRUE signal (logical 1), means that the input or output is activated.

The 8 bits in the HandShake register have the following meaning:



RI Ring Indicator. The signal level on this input has no effect on the operation of the transmitter or receiver circuits.

DTR Data Terminal Ready. A TRUE signal on this output indicates that the PD 3940 is switched on and the interface circuit is ready for communication. If the InputBuffer is full, the signal level is set to FALSE.

RTS Request To Send. The PD 3940 module holds this output TRUE, when data within an element from the OutputBuffer is being transmitted.

CTS Clear To Send. The signal level on this input must be TRUE, to enable the PD 3940 to start transmitting an element from the OutputBuffer.

DSR Data Set Ready. A TRUE signal on this input, indicates that the external equipment is connected to the PD 3940 module and ready to receive data. If data is being sent and the signal changes to FALSE, transmission stops when the current byte has been sent.

DCD Data Carrier Detect. The signal level on this input has no effect on the operation on the transmitter or receiver circuits.

SWNo $16: BaudRate

This variable selects the baud rate for the communication port. If an illegal baud rate is selected, i.e the baud rate is not implemented, an error code is generated. BaudRate can hold the following values:

76800, 38400, 19200, 9600, 4800, 2400, 1200, 600, 300

SWNo $18: ReserveTimer

ReserveTimer is automatically preset with the value from ReservePreset (SWNo $1A), each time the channel is reserved (Reservation boolean set TRUE by TestAndSet), and each time a string is sent to the OutputBuffer. If the ReserveTimer reaches zero before it is preset again, the Reservation boolean will be cleared automatically. The ReserveTimer may also be preset by direct access.

SWNo $19: ChConfig

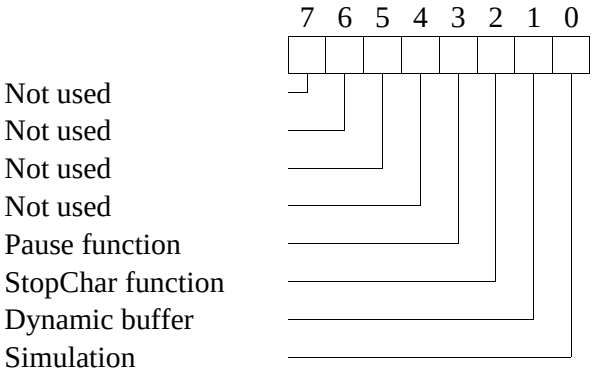
This variable selects the mode of operation for the channel and the parity for the data on the RS232 port.

The ChConfig variable is a record having following structure:

```
Record
    Enablebit   : Bit8;          (* Offset = 0 *)
    Functions   : BYTE;          (* Offset = 1 *)
    Ref_A       : BYTE;          (* Offset = 2 *)
    Ref_B       : BYTE;          (* Offset = 3 *)
end
```

where each field can be interpreted as follows:

Enablebit:



The EnableBits: Pause, StopChar and Dynamic, are all related to DatamodeIn, and refer to the values in the DatamodeParam register.

If "Dynamic buffer element sizing" is enabled (ChConfig.Enablebit[1] = TRUE), then reading the InputBuffer will access all bytes received up until the time of the read operation. After this, receipt of subsequent bytes will be stored in a new buffer element. When using "Dynamic buffer element sizing", only normal read operations of up to 56 bytes are supported. LongLoad (more than 56 bytes) is not allowed.

If an attempt is made to assert an Enablebit for a non-supported facility, a configuration error will be generated in ChError.

If Simulation is selected, writing to the InputBuffer is legal via P-NET. Data from the port will be ignored.

Functions:
The Function field holds the protocol of the RS 232 port:

Functions = $00 => Channel disabled
Functions = $03 => DatamodeIn
Functions = $04 => DatamodeOut
Functions = $05 => DatamodeInOut

DatamodeIn: The RS232 port is configured for input mode only, and data from external equipment can be received into the InputBuffer.

DatamodeOut: The RS232 port is configured for output mode only, and data to external equipment can be sent from the OutputBuffer.

DatamodeInOut: The RS232 port is configured for both input mode and output mode. Data from external equipment can be received into the InputBuffer and data to external equipment can be sent from the OutputBuffer.

Ref_A:
The Ref_A field holds the parity of the Port:

Ref_A = $00 => None
Ref_A = $02 => Even
Ref_A = $03 => Odd
Ref_A = $04 => Mark
Ref_A = $05 => Space

Ref_B is not used.

Every time a write operation is performed to either BaudRate (SWNo $16) or ChConfig (SWNo $19), the corresponding port is initialised, using the current contents of these variables.

SWNo $1A: ReservePreset
The maximum time period that the OutputBuffer can be reserved, is defined in seconds, in this register.

SWNo $1B: Reservation
This boolean variable is used to reserve the communication channel. The reservation facility is used in conjunction with DatamodeOut. Reservation = TRUE, indicates that the communication channel is reserved. The variable must be set TRUE with the TestAndSet function, in order to reserve the communication channel. For more information about the TestAndSet function, refer to the P-NET Standard ref.no. 502058.

A TestAndSet access to the Reservation boolean, will preset the ReserveTimer (SWNo $18) with a value found in ReservePreset (SWNo $1A). When the OutputBuffer is no longer required, it must be released again, by setting Reservation = FALSE.

If the ReserveTimer reaches zero before it is preset again, (by writing to OutputBuffer), the Reservation boolean will be cleared, and a Reservation Timeout error code will be generated in the ChError register.

SWNo $1C: DatamodeParam
This variable holds parameters concerning DatamodeIn only.

```
Record
    Pause          : REAL;      (* Offset = 0 *)
    ElementSize    : BYTE;      (* Offset = 4 *)
    StopChar       : CHAR;      (* Offset = 5 *)
end
```

The combination of ChConfig and this variable, determines how data is transferred to the Input-Buffer.

Pause:
If there has been a break in the transmissions to the RS232 port in the PD3940, which exceeds the time period defined in the Pause register, the current element in the InputBuffer will be considered to be complete, and subsequent data will be transferred to the next buffer element. This function is enabled when ChConfig.Enablebit[3] = TRUE.

ElementSize:
When the number of bytes defined in ElementSize have been received, the current element in the InputBuffer will be considered to be complete, and subsequent data will be transferred to the next buffer element. This function is enabled when ChConfig.Enablebit[1] = TRUE.

StopChar:
When a character equal to StopChar is received, the current element in the InputBuffer will be considered to be complete, and subsequent data will be transferred to the next buffer element. This function is enabled when ChConfig.Enablebit[2] = TRUE.

Each byte received, is in fact, transferred directly to the InputBuffer. If the InputBuffer is full when a byte is received, an InputBuffer Overrun error code is generated in the ChError register.

When the first byte for a new buffer element is received, the buffer element is automatically "reserved". The received byte is inserted in the buffer element, and the length parameter of the string is set to 1. As other bytes are received, they are now inserted into the reserved element, and the length parameter of the string is incremented. This continues until one of the conditions for transferring data to the next buffer element is fulfilled. When this occurs, the "Number of elements" register is incremented by 1. A description of the registers "Number of free elements" and "Number of elements" can be found, under the Complex Variable - Buffer section, in the P-NET Standard (reference no. 502058).

SWNo $1D: Maintenance
The Maintenance variable is used for service management and maintenance purposes, and holds the last date of service and an indication of the type of service.

```
Record
    Date      : BYTE;        (* Offset = 0 *)
    Month     : BYTE;        (* Offset = 1 *)
    Year      : BYTE;        (* Offset = 2 *)
    Type      : BYTE;        (* Offset = 3 *)
end
```

SWNo $1E: ChType
For the Communication Channel, ChType has the following form:

```
Record
    ChannelType: WORD;       (* Offset = 0 *)
    Exist: Bit16;            (* Offset = 2 *)
    Functions: Bit16;        (* Offset = 4 *)
end
```

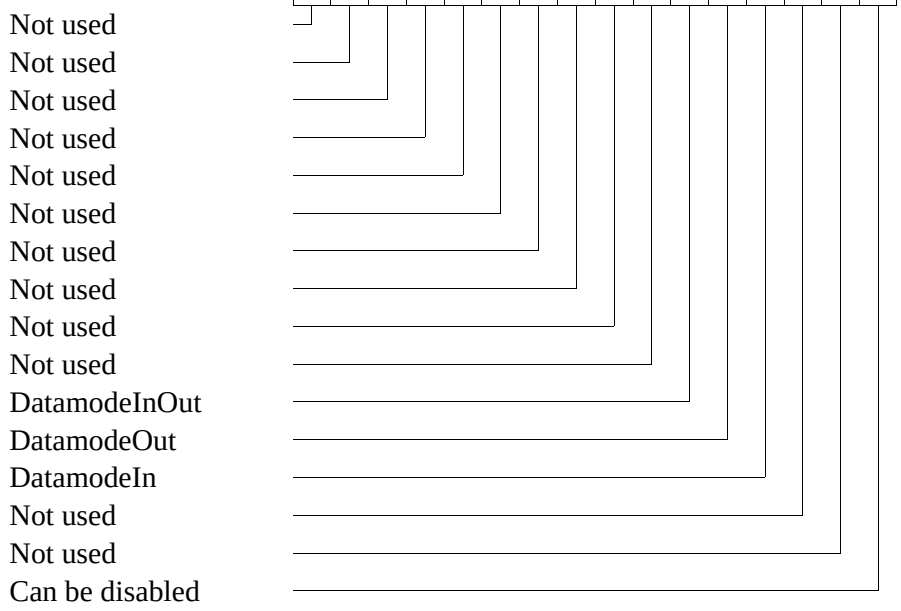
For the communication channel, ChType has the following value:
ChannelType = 14

Exist =

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	0	1	0	0	0	1	1	1

Functions =

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1

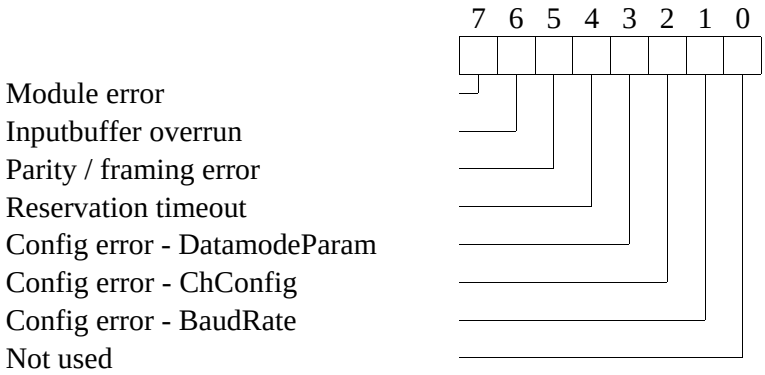


When selection of un-supported Baudrates, Parity modes, protocols etc. is attempted, configuration errors will be generated in ChError.

SWNo 1F: ChError

```
Record
    His:Array[0..7] of Boolean;     (* Offset = 0 *)
    Act:Array[0..7] of Boolean;     (* Offset = 2 *)
End
```

The 8 bits in ChError.His and ChError.Act have the following meaning. When an error occurs, the corresponding bit is set in both ChError.His and ChError.Act. When the error disappears, the bit is cleared in ChError.Act.



Bit 7 **Module error.** If this bit is set, the rest of the bits have no meaning, because a module error can cause random error codes within the individual channels (see also "Service channel").

Bit 6 **InputBuffer overrun** error is generated when the transfer of bytes to the InputBuffer is not possible, because the buffer is full. The error is cleared in ChError.Act by reading the InputBuffer.

Bit 5 **Parity/framing error** is generated if a parity or framing error is detected within any of the received data for the InputBuffer. The error is cleared in ChError.Act, when the Input-Buffer is empty.

Bit 4 **Reservation Timeout** is generated if the Reservation timer reaches zero. Consequently, the Reservation boolean is cleared. The error is cleared in ChError.Act, when the Reservation variable is set TRUE.

Bit 3 **Config error - DatamodeParam** is generated when an illegal configuration of the DatamodeParam variable is attempted. The error is generated in ChError.His and in ChError.Act. The error is cleared in ChError.Act by writing a legal configuration to the DatamodeParam variable.

Bit 2 **Config error - ChConfig** is generated when an illegal configuration of the ChConfig variable is attempted. The error is generated in ChError.His and in ChError.Act. The error is cleared in ChError.Act by a writing a legal configuration to the ChConfig variable.

Bit 1 **Config error - BaudRate** is generated when an illegal BaudRate selection is attempted. The error is generated in ChError.His and in ChError.Act. The error is cleared in ChError.Act by writing a legal BaudRate value to the BaudRate variable.

4 Construction, Mechanical

The PD 3940 module is housed in a grey plastic case. The case measures W x H x D = 122.0 x 120.0 x 55.0 mm (tolerance to DIN 16901).

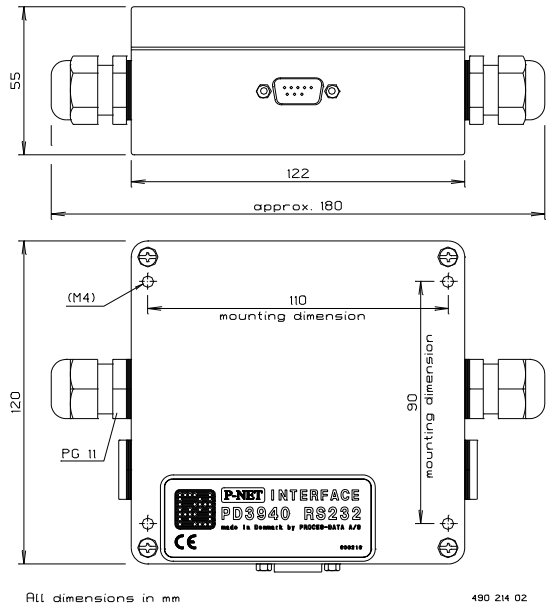
The module includes a snap connector, which provides terminals for field connection, power and communications. It has a built in RS232 9-pin male sub-D connector. The PD 3940 is mounted in an IP53 polycarbonate mounting box. Four PG11 cable glands are provided, for connection of the P-NET fieldbus and power supply cables.

Scale drawing (in mm):

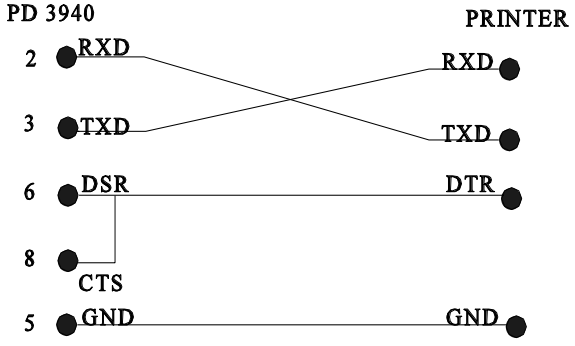
Materials

Case : Grey Polycarbonate (IP 53)

Weight : 500 gram.



5 RS 232 connection.



6 Specifications

All electrical characteristics are valid at an ambient temperature of -25 °C to +70 °C, unless otherwise stated.

All specifications apply within the approved EMI conditions.

6.1 Power supply

Power supply DC:	nom.	24.0 V
	min.	20.0 V
	max.	28.0 V
Ripple :	max.	5 %
Power consumption :	max.	1.2 W
Current at power up :	max.	200 mA

Fuse 300 mA (automatic).

6.2 RS232 Communication Interface:

Baudrates:	300 to 76,800
Parity:	None, even, odd, mark, space
Output buffer:	*Buffer[10] of STRING[255]*
Input buffer:	*Buffer[10] of STRING[255]*

6.3 Ambient Temperature

Operating temperature :	-25 °C to +70 °C
Storage temperature :	-40 °C to +85 °C

6.4 Humidity

Relative humidity :	max.	95 %

6.5 Approvals

Compliance with EMC-directive no.:	89/336/EEC
Generic standards for emission:	
Residental, commercial and light industry	EN 50081-1
Industry	EN 50081-2
Generic standards for immunity:	
Residental, commercial and light industry	EN 50082-1
Industry	EN 50082-2
Vibration (sinusoidal):	IEC 68-2-6 Test Fc

7 Survey of variables in the PD 3940 module

SWNo	**Service** 0	**Port_1** 1
x0	NumberOfSWNo	OutputBuffer
x1	DeviceID	InputBuffer
x2		HandShake
x3	Reset	
x4	PnetSerialNo	
x5		
x6		BaudRate
x7	FreeRunTimer	
x8		ReserveTimer
x9	ModuleConfig	ChConfig
xA		ReservePreset
xB		Reservation
xC		DatamodeParam
xD	WriteEnable	Maintenance
xE	ChType	ChType
xF	CommonError	ChError