502 069 02

POWER MONITOR MODULE

PD 3260 PMM

Manual

GB *February 1997*

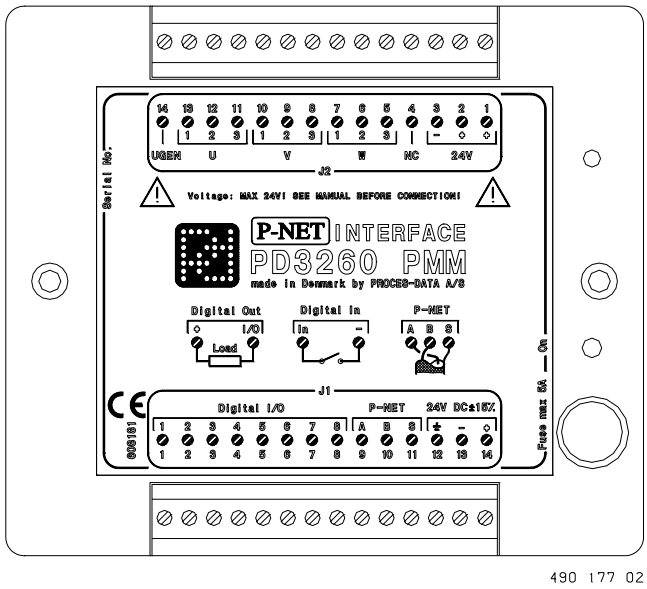
PROCES-DATA A/S, Navervej 8, 8600 Silkeborg, Denmark, Phone +45-87 200 300, fax + 45-87 200 301

Contents

1 General information.

The PD 3260 Power Monitor is a member of the 3000 series of modules from PROCES-DATA.

It provides a three-phase Power Line Monitor and Generator Switch Controller interface to the P-NET Fieldbus EN50170 Vol 1.



The monitoring functions offered by the module are useful for assessing the operation of electric generators, and for monitoring loads connected to three-phase power lines, or those having up to 3 single phases.

The generator switch functions are designed to perform phase, speed and acceleration controlled switching of synchronous or asynchronous generators, connected to three-phase power lines.

Switching is performed using the Digital Output Channels or via the Thyristor Switch Channel.

P-NET is a fieldbus network, designed for process control and data collection. The PD 3260 module has been developed to directly interface with digital process signals.

Configuration of the module for the functions required, and communication between the module and a control computer, is carried out via the P-NET fieldbus. The PD 3260 module can be controlled via P-NET, or it can operate as an autonomous unit.

The module possesses a programmable Calculator, which can be purpose programmed to control the digital outputs and monitor the digital inputs. The Calculator can operate on a variety of variable types, such as reals, integers, bytes, booleans, timers and arrays. User programmes can be devised to produce application specific functions, such as digital control loops, and PLC functions may be set up for use in a wide variety of local autonomous process applications.

The compact design and outstanding environmental specifications for the Power Monitor, makes it an ideal process component in industrial, as well as less stringent environments.

1.1 Features.

- Power Monitor Channel with three-phase current and voltage inputs.
- Generator Switch Channel with tachometer and voltage inputs.
- Thyristor Switch Channel with six modulated outputs for a three-phase thyristor switch.
- 8 Digital I/O Channels with automatic output functions.
- Output current measurement and overload protection.
- Common I/O Channel.
- Programmable Calculator and Internal Data Channel.
- Continuous self test.
- Watch Dog Timer.
- P-NET Fieldbus Communication (EN50170, Vol. 1).
- Rail mounting module (DIN / EN).
- EMC approved (89/336/EEC).

1.2 System description.

Power Monitor Channel
The Power Monitor channel has current and voltage inputs for the monitoring of a three-phase power line. RMS current, RMS voltage, Power factor, Power and Energy are continuously measured for each of the three phases. Input overload alarms, for the current and voltage inputs, can be enabled.

Total effective power is calculated, and the level can be monitored by enabling user presettable HighLevel and LowLevel alarms. In addition, the total effective power is integrated over time, to provide Total, Produced and Consumed Energy values. The line frequency is continuously measured.

Generator Switch Channel
The Generator Switch channel can be used to connect accelerating synchronous or asynchronous generators to a power line. The channel measures generator phase, speed and acceleration in order to calculate the moment in time when the generator should be connected to the power line. The generator switch channel can be used to control a digital output channel driving a circuit breaker, or the Thyristor Switch Channel. A configurable switch-on period value can be used to compensate for relay contact delay.

Thyristor Switch Channel
The Thyristor Switch Channel performs ramp controlled thyristor triggering for three-phase thyristor switches. The digital outputs I/O 1 to I/O 6 are used as the controlled outputs for this channel.

Digital I/O Channels:
Various automatic functions can be selected within each digital channel, such as automatic feedback control (single, as well as double), one-shot output and pulse output. This reduces such basic operations having to be performed at a remote central control point.

The output current (Sink), is measured continuously on each channel, and can be read as a value in Amps. If the current exceeds the specified max value, the output is switched off, and an error code (overload) is generated in the module.

The common I/O channel in the module, provides the ability to read/set all inputs/outputs, using a single P-NET transmission.

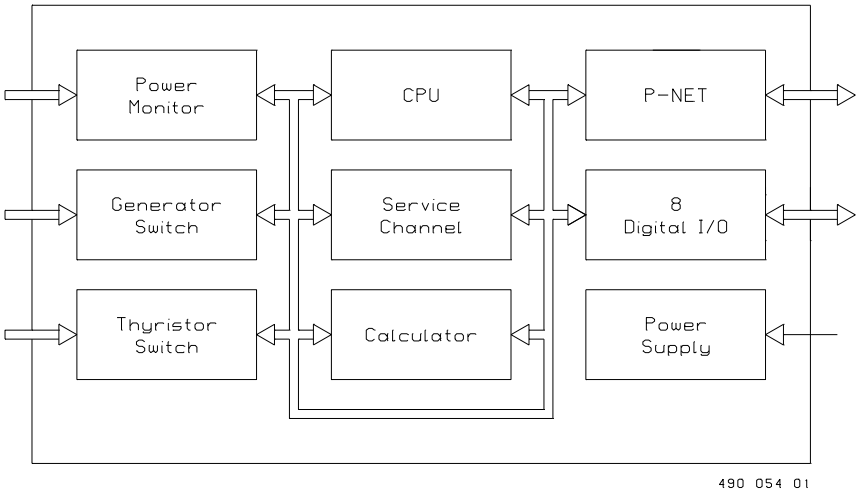
General
The unit offers comprehensive self-testing features, which enables the reporting of disconnection, overload and process failure. All outputs are protected against overload.

The watchdog timer ensures the safe shut down of a process following a communication error or power failure.

The PD 3260 is approved in compliance with the **EMC-directive no 89/336/EEC**. Test limits are determined by the generic standards **EN 50081-1** for emission and **EN 50082-2** for immunity.
The PD 3260 is approved in compliance with the **IEC 68-2-6 Test Fc** standard for vibration.

1.3 Channels/registers.

The PD 3260 module contains:



1 Service channel	(channel $00)	8 Digital I/O`s	(channel $01-08)
1 Common I/O channel	(channel $09)	1 Power Monitor	(channel $0A)
1 Generator Switch	(channel $0B)	1 Thyristor Switch	(channel $0C)
1 Program channel	(channel $0D)	1 Data channel	(channel $0E)

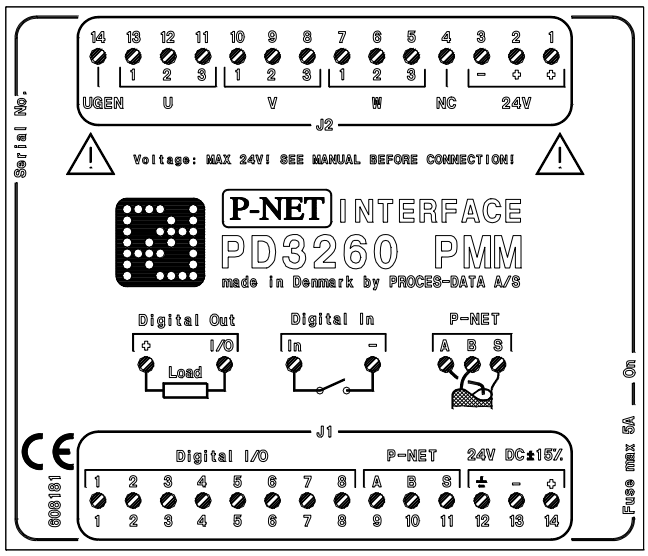
A set of 16 variables, numbered from 00 - $0F, is associated with each channel. For addressing a variable within a particular channel, a logical address called a SoftWire Number (SWNo), is used. The SWNo is calculated as: (channel number $*$ $10 + variable number within the channel).

Example: Variable 4 on channel 3 needs to be addressed.
The SWNo will therefore be $34.

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

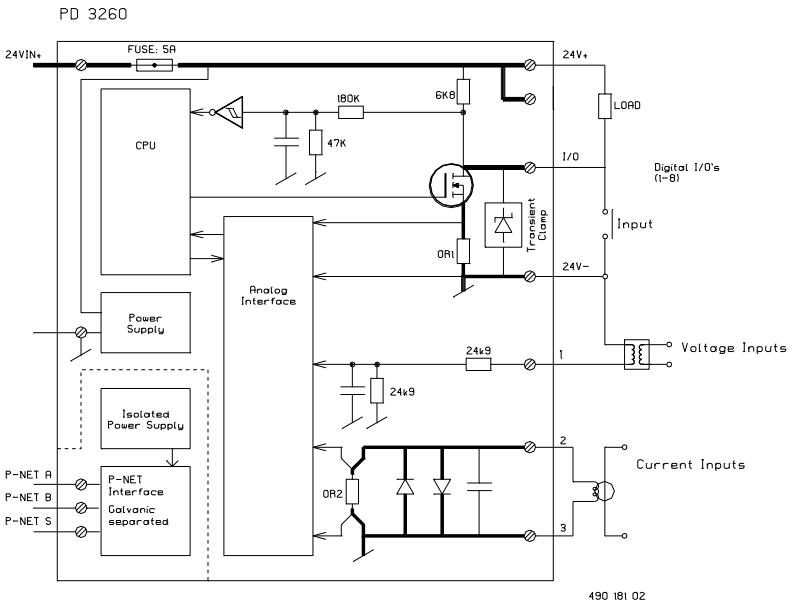
1.4 Connections.

The PD 3260 Power Monitor is physically designed as a black box, having two 14 pin connectors with screw terminals. The connectors are removable and equipped with a key pin, to avoid reversed connections. Connection identities are printed on the top of the module. The module wiring should be designed with a maximum of 2 wire connections in each screw terminal.
The module has a built in fuse, which is used to protect the module, and externally connected wiring and equipment. When connecting external equipment, the additional fuse protected +24 V screw terminals should be used (see hardware diagram).



490 179 02

Hardware diagram, principle.



490 181 02

1.5 Memory types.

The PD 3260 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 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 modified via P-NET.

RAM AutoSave

Has the same function as RAM InitEEPROM, with the addition that the contents of RAM are automatically copied to EEPROM, at a frequency of approximately 10 hours.

After copying to EEPROM, a new sumcheck value is calculated. The new sumcheck value is stored in the EEPROM, unless an EEPROM error has already been set (CommonError.ChError.Act[2]).

2 Service Channel (channel 0).

PD 3260 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	WDTimer	RAM Read Write	Decimal	Real
9	ModuleConfig	EEPROM RPW	- - - - - - -	Record
A	WDPreset	EEPROM RPW	Decimal	Real
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 is of the following type:

```
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 = 3260
ProgramVersion= 100 (the first version)
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; (* Node Address *)     (* Offset = 0 *)
    SerialNo: String[20];                (* Offset = 2 *)
end
```

The serial number is used for service purposes and as a 'key' to 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 the 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 serial number, and PD is the initials of PROCES-DATA.

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

P-NET Watch Dog function
PD 3260 Digital I/O is equipped with a P-NET Watchdog, which switches off all the digital outputs, by clearing OutFlags and Control flags, if P-NET communication ceases. The P-NET watchdog uses SWNo 8 and SWNo $A.

SWNo 8: WDTimer [s]
WDTimer is automatically preset with the value from WDPreset (SWNo $A), either each time the module is called via P-NET, or following a power-up or module reset. If the WDTimer reaches zero before it is preset again, the PnetWDRunOut flag will be set, and all the outputs will switch OFF. The timer contains a value in sec.

SWNo 9: ModuleConfig
The 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
```

The EnableBit, Ref_A and Ref_B fields are not utilised in the Service channel.

The watchdog facility may be switched on and off by means of the field variable Functions, as shown below.

ModuleConfig.Functions = 0	Watchdog
ModuleConfig.Functions = $10	No watchdog

SWNo $A: WDPreset [s]
The maximum time allowable between two calls to the module, before the watchdog is activated, is defined in this register, in seconds.

SWNo $D: WriteEnable
Write protected variables can only be changed when WriteEnable is TRUE. After 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". This sum check calculation period is approximately 0.25 second. Consequently, the module should not be reset during this period, otherwise an EEPROM error can occur (see SWNo $F: CommonError).

The WriteEnable variable may be optionally write protected, by means of a hardware lock.
If the hardware switch, marked “2" inside the module, is set to “ON”, the value in WriteEnable becomes write protected. NB: Disconnect module power before opening the module. Using this

facility, and mechanically sealing the module housing, will ensure that all EEPROM variables in the module are permanently write protected,

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, the contents of which depends on the channel type.

The record for the service channel has the following structure:

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

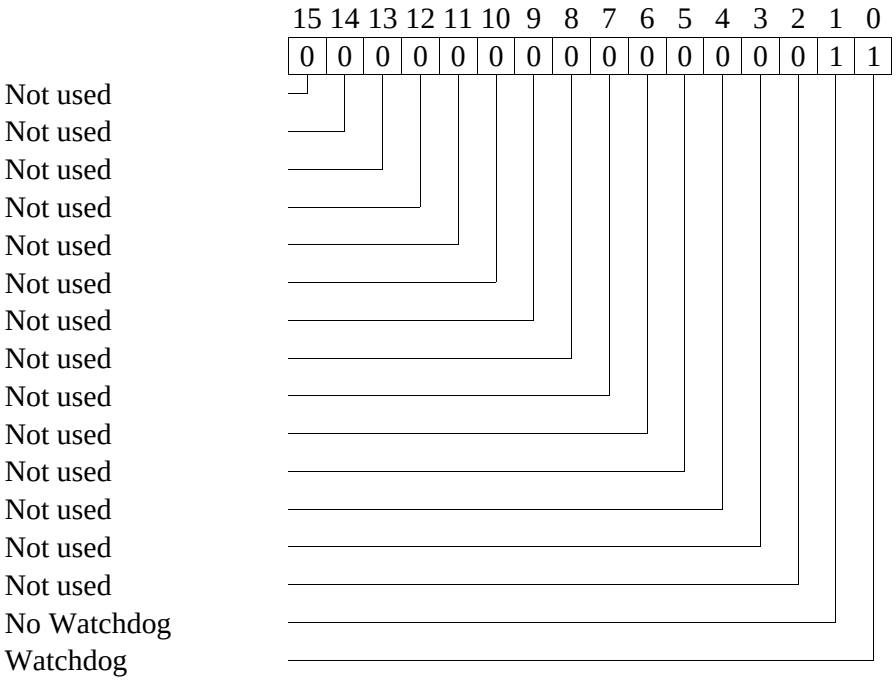
For the service channel, ChType has the following value:

ChannelType = 1

Exist =

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

Functions =

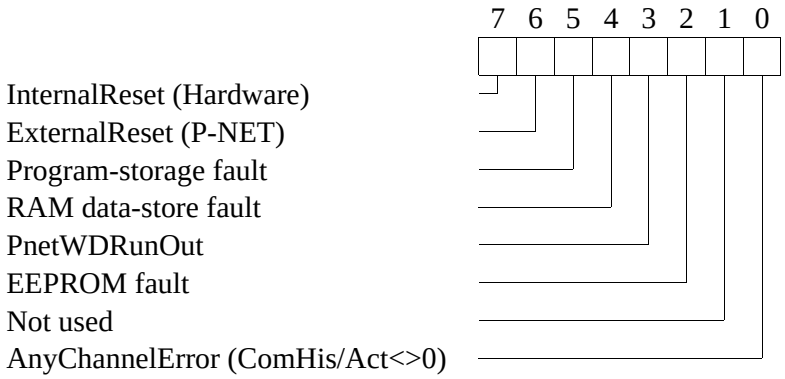


SWNo $F: CommonError

The CommonError variable holds error information on 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 3 PnetWDRunOut is set TRUE, if the WDTimer reaches zero and the Watchdog function is switched ON.

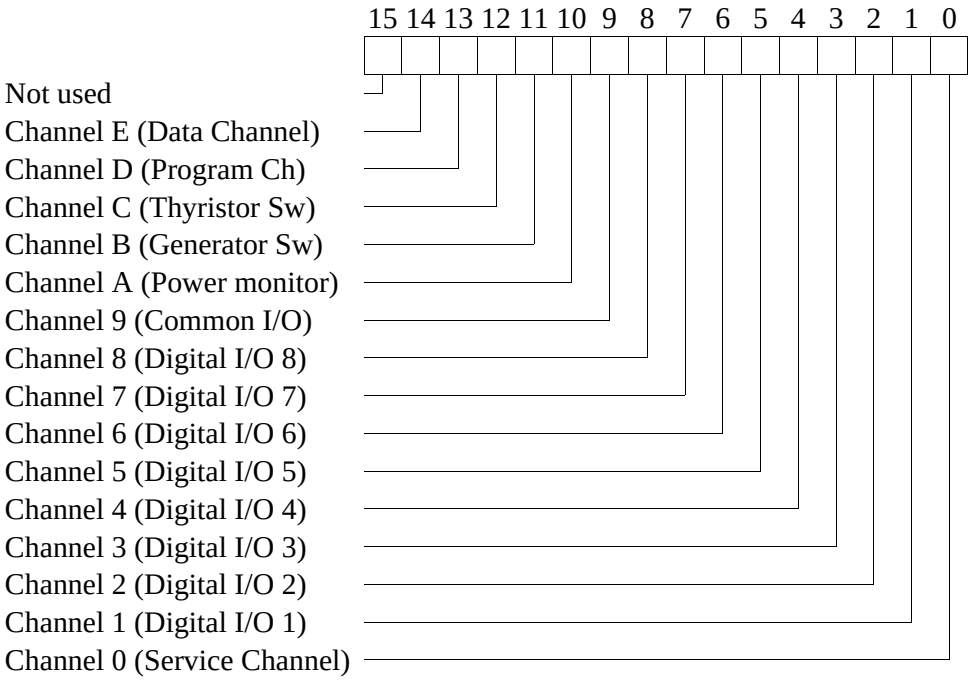
Bit 2 EEPROM fault is set to TRUE, if the self test finds an error in the data memory (EEPROM). The error may be corrected by setting and resetting WriteEnable.

Bit 0 AnyChannelError = 1, means that an error, or an unacknowledged 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.

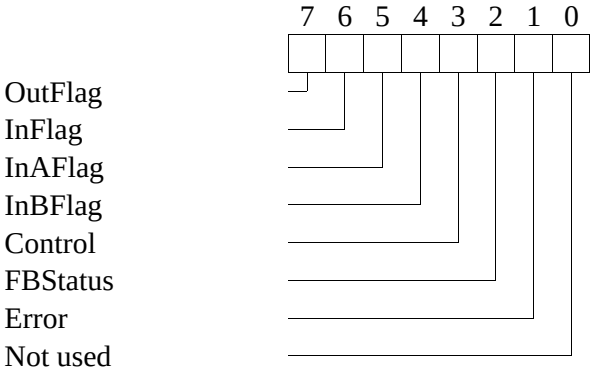

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Not used
Channel E (Data Channel)
Channel D (Program Ch)
Channel C (Thyristor Sw)
Channel B (Generator Sw)
Channel A (Power monitor)
Channel 9 (Common I/O)
Channel 8 (Digital I/O 8)
Channel 7 (Digital I/O 7)
Channel 6 (Digital I/O 6)
Channel 5 (Digital I/O 5)
Channel 4 (Digital I/O 4)
Channel 3 (Digital I/O 3)
Channel 2 (Digital I/O 2)
Channel 1 (Digital I/O 1)
Channel 0 (Service Channel)

3 Digital I/O Channel (channel 1 - 8).

Variables on digital I/O Channel x. Channel identifier: **Digital_IO_x**

SWNo	Identifier	Memory type	Read out	Type	SI Unit
x0	FlagReg	RAM Read Write	Binary	Bit8	- - -
x1	OutTimer	RAM Read Write	Decimal	Real	s
x2	Counter	RAM AutoSave	Decimal	Long integer	- - -
x3	OutCurrent	RAM Read Write	Decimal	Real	A
x4	OperatingTime	RAM AutoSave	Decimal	Real	s
x5	UserByteArray	RAM Init EEPROM	- - - - -	Array4Byte	- - -
x6	FBTimer	RAM Read Write	Decimal	Real	s
x7	FB Preset	EEPROM RPW	Decimal	Real	s
x8	OutPreset	EEPROM RPW	Decimal	Real	s
x9	ChConfig	EEPROM RPW	- - - - -	Record	- - -
xA	MinCurrent	EEPROM RPW	Decimal	Real	A
xB	MaxCurrent	EEPROM RPW	Decimal	Real	A
xC	UserRealArray	EEPROM RPW	- - - - -	Array2Real	- - -
xD	Maintenance	EEPROM RPW	- - - - -	Record	- - -
xE	ChType	PROM Read Only	- - - - -	Record	- - -
xF	ChError	RAM Read Only	Binary	Record	- - -

SWNo x0: FlagReg



Bit 7: OutFlag

This flag controls the output, if the P-NET WDRunOut bit is FALSE, and the channel is configured as an output. The special output functions control the output flag, in the same way as a P-NET transmission.

OutFlag:=True => Output ON
PnetWDRunOut = True => OutFlag:=False => Output OFF

Bit 6: InFlag
The input flag is controlled by the input detector, and shows the logic level on the input terminals. The input flag is true, when the input is connected to 24V-. If the channel is used as an output, the input flag will follow the output flag. If the output terminals are short-circuited, the input flag will not follow the output flag. The input signal can be simulated, by setting the channel in input simulation mode (ChConfig.Enablebit[0] = TRUE), and subsequently writing the state to the input flag. The input flag is FALSE after reset in simulation mode.

Feedback Control
Many process components are equipped with feedback contacts. A typical example is a valve with 1 or 2 micro switches, which indicate the mechanical position of the piston. In this application, in addition to the output, one or two inputs are needed. The inputs to be used by the feedback-control, are selected in ChConfig.Ref_A and ChConfig.Ref_B in the output-Channel.

Bit 5 and Bit 4: InAFlag, InBFlag
The InAFlag is identical to the input flag for the channel selected as Feedback input A. Feedback input A is the input signal which should correspond with the same state as the output signal. Therefore, Feedback input A must be TRUE when the output is TRUE, to indicate a correct feedback signal. The Feedback input A channel is selected in ChConfig.Ref_A. The InBFlag is identical to the input flag for the channel selected as Feedback input B. Feedback input B is the input signal which should correspond with the inverse state of the output signal. This input signal is selected in ChConfig.Ref_B. The feedback signals can be simulated, by setting ChConfig.Enablebit[2] = TRUE. If feedback simulation is selected, the InAFlag and InBFlag are automatically set to the correct state, to correspond with the current state of OutFlag.

Bit 3: Control
When the Control flag is set, a special output function for the channel is enabled, (one shot output, 50 % duty cycle output or timer output), which then controls the output. Clearing the Control flag will disable the special output function, and clears the output, which may then be controlled via P-NET. After a power-up or a reset of the module, the control flag is set FALSE, unless the channel is configured to be controlled by the Thyristor Switch Channel, in which case, the Control flag is always set TRUE. The state of the Control flag has no influence on the output if the channel is configured as an output without a special output function.

Bit 2: FBStatus
The FBStatus indicates the current feedback condition. The value of FBStatus does not depend on the FBTimer, which means that the actual valve position for example, can be ascertained as correct, or incorrect, before the FBTimer has reached zero. If feedback is used (single or double), FBStatus is always set TRUE when the OutFlag is changed. If no feedback is used, FBStatus always reads FALSE.
FBStatus = TRUE, indicates that the feedback signal/s are incorrect.

Bit 1: Error
The purpose of this Error bit is to indicate an error condition on the channel, (incorrect feedback signals, overload, underload, PrgError and hardware errors).
Error = TRUE, indicates ChError.Act <> 0. (See SWNo xF).

SWNo x1: OutTimer [s]
Each output channel has a timer, used with the special output functions. The timer is either preset via P-NET, or from the preset register, depending on which function is selected for the channel. The timer counts down, with a resolution of 1/32 second. The count continues through negative values. The timer register is cleared after a power failure. The maximum value for the timer is approximately 97 days. Following an overflow, the timer continues from it's maximum value.

SWNo x2: Counter
The counter counts the number of pulses at the input. The maximum count frequency is 50 Hz. The counter counts up to a maximum of 2147483647 (a LongInteger).
When the counter exceeds +2147483647, it restarts at -2147483648. The counter increments by one, every time the InFlag changes from "0" to "1".

SWNo x3: OutCurrent [A]
The OutCurrent register indicates the sink current in the output load. It is measured with a resolution of approximately 12 mA. The stability of the measurement depends on the power supply stability.

SWNo x4: Operatingtime [s]
This variable totalises the time period when InFlag is True. The resolution for OperatingTime is 0.125 sec.

SWNo x5: UserByteArray
The variable UserByteArray has the type Array[1..4] of Byte, and may be used for temporary values or special configuration parameters, particularly in connection with a calculator program. The variable has no effect on the standard functions for a digital I/O channel.

SWNo x6: FBTimer [s]
The FeedBack-timer is used to disable feedback error detection while the mechanical components are changing position. The FBTimer is preset from FBPreset when the output changes state. The timer counts down.

```
If FBTimer < 0 then
begin
  ChError.Act[FeedbackError] := FlagReg[FBStatus];
  ChError.His[FeedbackError] := FlagReg[FBStatus];
end
ELSE ChError.Act[FeedbackError] :=False.
```

SWNo x7: FBPreset [s]
This register holds a value equal to the maximum permitted time for incorrect feedback signals to be present, before an error is flagged. The value is passed to FBTimer when the output changes state.

SWNo x8: OutPreset [s]
This variable holds a preset value for the OutTimer. The preset value is passed to the OutTimer by the special output functions.

SWNo x9: ChConfig
This variable selects the I/O type, the type of feedback control (single or double feedback) and a choice of special output functions.

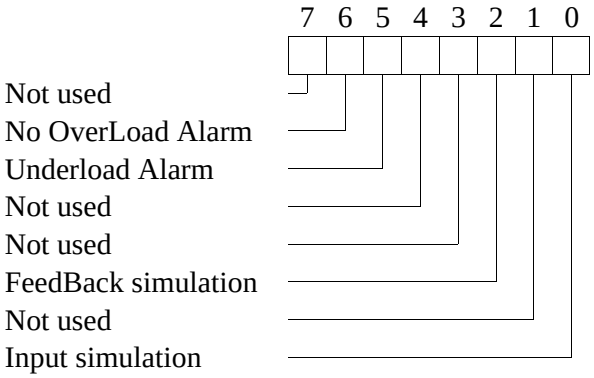
Feedback control: The correct feedback state is: Input A = Output AND Input B = NOT Output. The feedback inputs can be disabled by writing a 0 as the channel number, in ChConfig.Ref_A and/or ChConfig.Ref_B fields, if only one or no input channels are required.

The ChConfig 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
```

where each field can be interpreted as follows:

Enablebit:



Functions :

	Functions = $00	=> Input only
	Functions = $10	=> Output
	Functions = $20	=> One shot output
	Functions = $30	=> 50% Duty-Cycle output
*	Functions = $40	=> Output controlled by Thyristor Switch Channel
	Functions = $50	=> Timer Output

* Not implemented in Ch7 and Ch8.

Ref_A : Channel No. for FeedBack input A (input = output).

Ref_B : Channel No. for FeedBack input B (input ≠ output).

Special output-functions:

One shot output.
This automatic function is selected by setting ChConfig.Functions = $20. When the Control Flag is changed from FALSE to TRUE, the output will be set true for a period equal to the value of OutPreset. The time can be varied by re-loading the OutTimer. Output is reset if the Control Flag is set to FALSE and the output may be controlled directly via P-NET.

Precise Function description:

```
After reset: InternalState:=False; FlagReg[Control]:=False;

Loop
    If NOT InternalState and FlagReg[Control] then (* Positive edge*)
      OutTimer:=OutPreset
    If InternalState and NOT FlagReg[Control] then (* Negative edge*)
      FlagReg[OutFlag]:=False;
    InternalState:=FlagReg[Control];
    If FlagReg[Control] then
      If OutTimer > 0 then
        FlagReg[OutFlag]:=true ELSE FlagReg[OutFlag]:=false
End
```

50% Duty-cycle Output.
This automatic function is selected by setting ChConfig.Functions = $30. If the Control Flag is ON, the output is inverted with a time interval equal to OutPreset. The time for a period (one OFF period and one ON period) is twice the value of the contents of OutPreset.
If the Control Flag is reset, the output switches OFF and may then be controlled via P-NET.

Precise Function description:

```
After reset: InternalState:=False; FlagReg.Control:=False;

Loop
  If InternalState=False and FlagReg[Control]=True then (* Positive edge*)
  Begin
    OutTimer:=OutPreset;
    FlagReg[OutFlag]:=True
  End;
  If InternalState=True and FlagReg[Control]=False then (* Negative edge*)
    FlagReg[OutFlag]:=False;
  InternalState:=FlagReg[Control];
  If FlagReg[Control]=True and OutTimer <= 0 then
  Begin
    FlagReg[OutFlag]:=NOT FlagReg[OutFlag];
    OutTimer:=OutPreset;
  End
End
```

Output controlled by Thyristor Switch Channel.
The digital outputs can be configured to be controlled by the thyristor switch channel. In this configuration, two, four or six outputs are used, to generate a ramp controlled signal for thyristor triggering. The output signals are timed to control a three-phase, six-output thyristor switch, if all six outputs are used.

This function is selected, by setting ChConfig.Functions=$40 in the individual I/O channels. The thyristor switch channel uses the digital outputs in pairs: I/O 1+2, I/O 3+4 and I/O 5+6. Both of the IO´s in a pair must be configured, otherwise none of them will be used. The digital I/O 7 and I/O 8 cannot be used by the thyristor switch channel.

When the output is controlled by the Thyristor Switch Channel, the output from the I/O terminal is a modulated 10kHz signal. Reading flags in the I/O channel at this frequency has no meaning, and for the same reason, the “InternalDisconnection” test is disabled, to avoid incorrect error codes being generated.

Timer output.
This automatic function is selected by setting ChConfig.Functions = $50. When the Control Flag is TRUE, the output will be set ON, if OutTimer is greater than zero. The on period may be varied by re-loading the OutTimer with a new value. The output is reset, if the Control Flag is set to FALSE, and the output may then be directly controlled via P-NET.

Precise Function description:

```
After reset: InternalState:=False; FlagReg[Control]:=False;

Loop
    If InternalState and NOT FlagReg[Control] then (* Negative edge*)
      FlagReg[OutFlag]:=False;
    InternalState:=FlagReg[Control];
    If FlagReg[Control] then
      If OutTimer > 0 then
        FlagReg[OutFlag]:=true ELSE FlagReg[OutFlag]:=false
End
```

SWNo xA: MinCurrent [A]

This variable defines the minimum permitted current in the load when the output is ON. If OutCurrent is less than MinCurrent when the output is ON and the FBTimer < 0, an error may be generated. Error-code generation is enabled by setting ChConfig.Enablebit[5] TRUE (underload alarm).

Precise Function description:

```
If (OutCurrent < MinCurrent) and (FlagReg.[OutFlag]=true)
      and (FBTimer < 0) and ChConfig.EnableBit[5] then
  ChError.Act[UnderLoad]:= true
else
  ChError.Act[UnderLoad]:= false
```

SWNo xB: MaxCurrent [A]

If Outcurrent exceeds MaxCurrent and FBTimer < 0, the output is switched off and an error is generated. For applications in which it is normal to let the max. current switch the output off, error code generation can be disabled, by setting ChConfig.Enablebit[6] TRUE (no overload alarm). MaxCurrent can not exceed 1.0 Amp. Any output current > 2 Amp switches the output off instantly.

Precise Function description:

```
If ((OutCurrent > MaxCurrent) and (FlagReg[OutFlag] = true) and (FBTimer < 0))
      or (OutCurrent > 2 Amp) then
Begin
 FlagReg[OutFlag]:=false;
 FlagReg[Control]:=false;
 If NOT ChConfig.EnableBit[6] then
 Begin
  ChError.Act[OverLoad]:= true;
  ChError.His[OverLoad]:= true;
 End
End
```

SWNo xC: UserRealArray

The variable UserByteArray has the type *Array[1..2] of Real* and may be used for special configuration parameters, particularly in connection with a calculator program. This variable has no effect on the standard functions for a digital I/O channel.

SWNo xD: 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.

The Maintenance is a record having the following structure:

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

SWNo xE: ChType

For the digital I/O channels, ChType has the following structure:

```
Record
      ChannelType: WORD;         (* Offset = 0 *)
      Exist: Bit16;              (* Offset = 2 *)
      Functions: Bit16;          (* Offset = 4 *)
      FeedBack: Bit16;           (* Offset = 6 *)
end
```

ChType has the following value:

ChannelType = 2

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	1	1	1	1	1	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	1	1	1

Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Timer Output
Thyristor Switch *
50% Duty-Cycle output
One shot output
Output
Input only

* Not implemented in Ch7 and Ch8

Feedback =

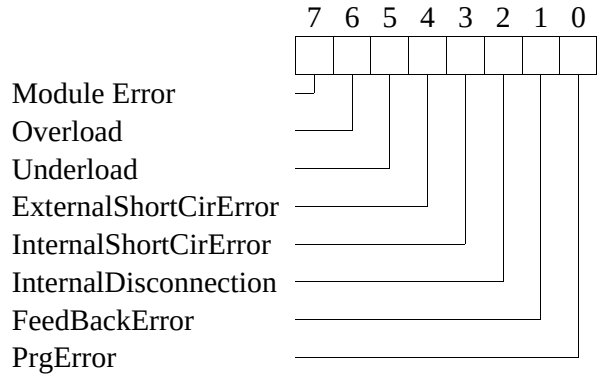
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Input = Output
Input <> Output

SWNo xF: ChError

```
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 on the individual channels (see also "Service channel").

Bit 6 OverLoad is set if the current in the output load exceeds MaxCurrent (default 1 A). The Overload alarm can be disabled by setting ChConfig.Enablebit[6] TRUE. ChError.Act[OverLoad] remains set until a **Write** operation is performed on the **FlagReg** variable.

Bit 5 UnderLoad is set if the load is disconnected (OutCurrent < MinCurrent). The Underload alarm can be enabled by setting ChConfig.Enablebit[5] TRUE.

Bit 4 ExternalShortcirError is set if an external short circuit error is detected (InFlag=1 and OutFlag=0). The error bit cannot be set on channels configured as input. This error bit will not appear in input simulation mode.

Bit 3 InternalShortCirError is set if the output transistor is short circuited (OutFlag=0 and OutCurrent > 0.1 Amp). This error bit will not appear in input simulation mode.

Bit 2 InternalDisconection is set, if the output transistor is disconnected (OutFlag=1 and InFlag = 0 and NOT overload). This error bit will not appear in input simulation mode or when the channel is controlled by the Thyristor Switch Channel.

Bit 1 FeedBackError is set if there is a feedback error (see FBTimer).

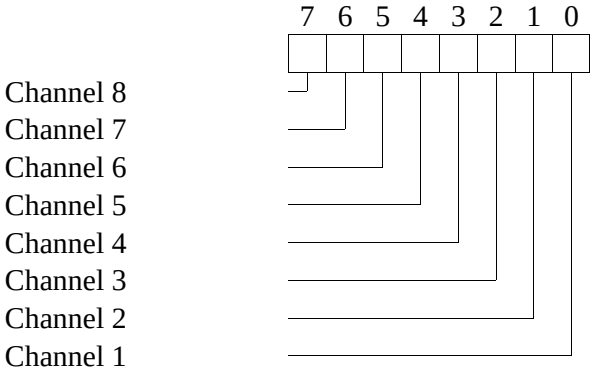
Bit 0 PrgError is set following attempts to set FlagReg[OutFlag], if the I/O is configured to be an input, or following attempts to set FlagReg[Control], if the I/O is configured to be an input or an output, without any automatic functions. ChError.Act[PrgError] remains set, until a **Write** operation is performed on the **FlagReg** variable.

4 Common I/O Channel (channel 9).

Variables on Common I/O channel. Channel identifier: **CommonIO**

SWNo	Identifier	Memory type	Read out	Type
90	OutFlags	RAM Read Write	Binary	Bit8
91	InFlags	RAM Read Write	Binary	Bit8
92	IOChError	RAM Read Only	Binary	Record
93				
94				
95				
96				
97				
98				
99				
9A				
9B				
9C				
9D				
9E	ChType	PROM Read Only	- - - - - -	Record
9F	ChError	RAM Read Only	Binary	Record

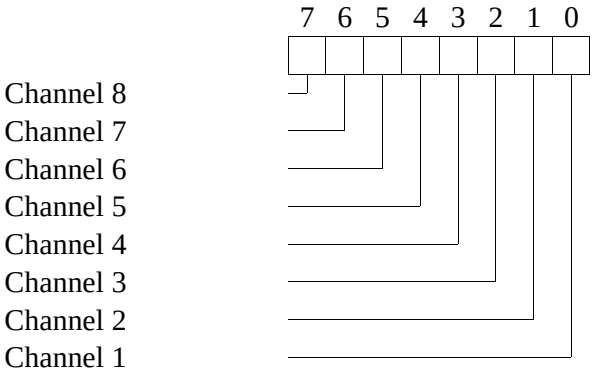
SWNo $90: OutFlags



This variable contains all the OutFlag's from all I/O channels. This means that all digital outputs in the module can be controlled from this register.

NOTE: When setting an outflag TRUE via the Common I/O channel, the FlagReg[FBStatus] will not be set TRUE, and the FBTimer will not be preset on the corresponding channels.

SWNo $91: InFlags



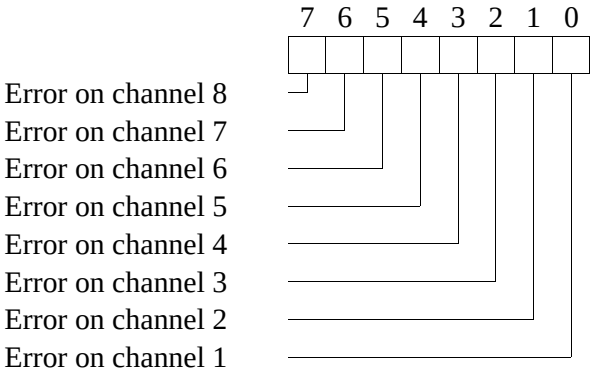
This variable contains all the InFlag's from all I/O channels. This means that all digital inputs in the module can be read in this register.

SWNo $92: IOChError

The IOChError is a variable of the following type:

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

Meaning of IOChError.His and IOChError.Act:



The IOChError variable indicates if there is an error, or an unacknowledged error, in one or more I/O channels.
IOChError.His is set if ChError.His of any channel <> 0, and IOChError.Act is set if any channel contains a ChError.Act <> 0.
Reading IOChError does not acknowledge the channel errors. This can only be performed by reading ChError in the individual channels, or by means of SWNo $F (channel 0).

SWNo $9E: ChType

For the common channel, ChType is of the following type:

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

The Common I/O Channel is a Company specific channel, where ChType has the following value:

ChannelType = $8009

Exist =

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

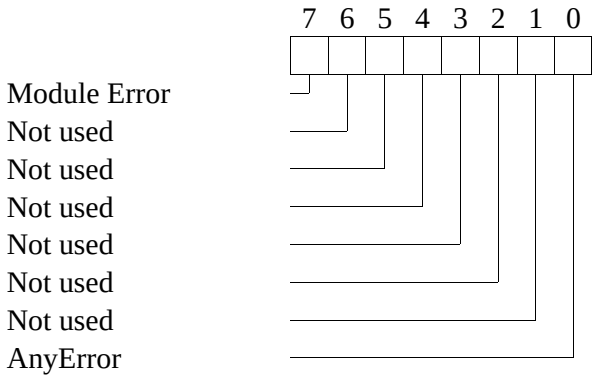
ExistingChannels =

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

SWNo $9F: CHError

```
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 on the individual channels (see also "Service channel").

Bit 0 AnyError is set if there is an error, or an unacknowledged error, in one or more I/O channels.
ChError.His[0] is set if any bit is set in IOChError.His, indicating any historical error on any I/O channel.
ChError.Act[0] is set if any bit is set in IOChError.Act indicating any actual error on any I/O channel.
Reading ChError does not acknowledge the channel errors. This can only be done by reading ChError in the individual channels, or by means of reading SWNo $F (channel 0).

5 Power Monitor Channel (channel A).

The power monitor channel has current and voltage inputs for monitoring a 50Hz or 60 Hz power line. The channel can be used as an energy meter. It measures the energy of one three-phase system, or between one and three single-phase systems. Another application of the device could be for monitoring the power factor of a plant and then compensating for this, by connecting a capacitor battery via the module's digital I/O channels.

The channel has 6 primary inputs for measuring three-phase RMS current and RMS voltage. Input overload alarms for the voltage and current inputs can be enabled. Using the primary inputs, the following parameters are calculated:
Power factor, Power and Energy are measured for each of the three phases, and as a total. The Total effective power level can be monitored, by enabling the user presettable HighLevel and LowLevel alarms. The total effective power is integrated over time, and presented as Total, Produced and Consumed Energy. The line frequency is also measured.

The primary inputs are precision factory calibrated. Automatic calculation of fullscale for voltage and current inputs are possible, to ease calibration of the module when using current and voltage transformers.

All variables are represented in engineering units. The power and energy variables can be individually scaled for the most appropriate units, by using the Factors register.

Variables on Power Monitor channel. Channel identifier: **PowerMonitor**

SWNo	Identifier	Memory type	Read out	Type	SI Unit
A0	Power	RAM Read Write	Decimal	Real	W
A1	Energy	RAM AutoSave	Decimal	Record	Wh
A2	PhasePower	RAM Read Write	Decimal	Record	W
A3	PhaseEnergy	RAM AutoSave	Decimal	Record	Wh
A4	Current	RAM Read Write	Decimal	Record	A
A5	Voltage	RAM Read Write	Decimal	Record	V
A6	PowerFactor	RAM Read Write	Decimal	Record	[]
A7	HighLevel	RAM Init EEPROM	Decimal	Real	W
A8	LowLevel	RAM Init EEPROM	Decimal	Real	W
A9	ChConfig	EEPROM RPW	- - -	Record	- - -
AA	LineFreq	RAM Read Write	Decimal	Real	Hz
AB	FullScale	EEPROM RPW	Decimal	Record	- - -
AC	Factors	EEPROM RPW	Decimal	Record	- - -
AD	Maintenance	EEPROM RPW	- - -	Record	- - -
AE	ChType	PROM Read Only	- - -	Record	- - -
AF	ChError	RAM Read Only	Binary	Record	- - -

SWNo $A0: Power.
The Power variable holds the total effective power. The calculation of the variable is dependent on the channel mode. In three-phase mode, the total power is calculated from the voltage and current inputs for the L1 and L2 lines. In single-phase mode, the total power is calculated as the sum of the three individual phases. See also section 5.1: Connections to Power Monitor Channel. The readout unit for Power may be scaled by the Factors.PowerScale variable.
HighLevel and LowLevel alarms can be enabled for the Power variable.

SWNo $A1: Energy.
The energy variable is a record having the following structure:

```
Record
    Tot: Real;   (* Total energy    *)      (* Offset = 0 *)
    Prod: Real;  (* Produced energy *)      (* Offset = 4 *)
    Cons: Real;  (* Consumed energy *)      (* Offset = 8 *)
End;
```

The first field in the energy record variable, holds the total power integrated over time. The second and third field in the variable, holds the integrated positive and negative power respectively, which could be used when a process is both producing and consuming power. The readout unit for Energy may be scaled using the Factors.EnergyScale variable. The energy variables can be cleared or preset, by writing new values to the variables.

SWNo $A2: PhasePower.
The PhasePower variable is only used when the channel is in single-phase mode. In this mode, the PhasePower variable holds the actual power at each phase. PhasePower is a record having the following structure:

```
Record
    U: Real;     (* Power at line L1 *)     (* Offset = 0 *)
    V: Real;     (* Power at line L2 *)     (* Offset = 4 *)
    W: Real;     (* Power at line L3 *)     (* Offset = 8 *)
End;
```

SWNo $A3: PhaseEnergy:
The PhaseEnergy variable is only used when the channel is in single-phase mode. In this mode, the PhaseEnergy variable holds the total energy in each phase. PhaseEnergy is a record having the following structure:

```
Record
    U: Real;     (* Energy from line L1 *)          (* Offset = 0 *)
    V: Real;     (* Energy from line L2 *)          (* Offset = 4 *)
    W: Real;     (* Energy from line L3 *)          (* Offset = 8 *)
End;
```

The phase energy variables can be cleared or preset, by writing new values to the variables.

SWNo $A4: Current.
The Current variable holds the measured currents from the three primary current inputs U2, V2 and W2. Current is a record having the following structure:

```
Record
    U: Real;    (* RMS-current line L1 *)    (* Offset = 0 *)
    V: Real;    (* RMS-current line L2 *)    (* Offset = 4 *)
    W: Real;    (* RMS-current line L3 *)    (* Offset = 8 *)
End;
```

The RMS currents are measured with current transformers in the lines L1, L2 and L3.
The common current scale factor must be inserted in FullScale.Current. Automatic calculation of FullScale.Current is performed if ServiceChannel.WriteEnable is true and the actual current measured with a reference instrument in L1 is written in the Current.U variable.

SWNo $A5: Voltage.
The Voltage variable holds either the measured line voltages (three-phase mode) or the phase voltages (single-phase mode) at the three primary voltage inputs U1, V1 and W1. Voltage is a record having the following structure:

```
Record
    U: Real;    (* RMS-voltage line/phase L1 *)    (* Offset = 0 *)
    V: Real;    (* RMS-voltage line/phase L2 *)    (* Offset = 4 *)
    W: Real;    (* RMS-voltage line/phase L3 *)    (* Offset = 8 *)
End;
```

The RMS voltages are measured with voltage transformers. The voltage scale factors must be inserted in the FullScale.VoltageU/V/W variables. Automatic calculation of the fullscale values is performed if ServiceChannel.WriteEnable is true and the actual voltages measured with a reference instrument is written in the corresponding Voltage.U, Voltage.V and Voltage.W variables.

SWNo $A6: PowerFactor.
The PowerFactor variable holds the total measured power factor. The power factor is defined as the cosine to the angle between voltage and current and is a measure for reactive (capacitive or inductive) loads. If the channel is in single-phase mode the variable also holds the measured power factors of each phase. In three-phase mode the power factor calculation assumes a balanced voltage system. Small differences in the line voltages (5-10%) however will produce fairly accurate results.

PowerFactor is a record having the following structure:

```
Record
    Tot: Real;   (* Total power factor *)      (* Offset = 0 *)
    U: Real;     (* Power factor line L1 *)    (* Offset = 4 *)
    V: Real;     (* Power factor line L2 *)    (* Offset = 8 *)
    W: Real;     (* Power factor line L3 *)    (* Offset = 12 *)
End;
```

SWNo $A7: HighLevel.

HighLevel is a limit switch for the Power variable having the following function:

IF Power > HighLevel AND ChConfig.Enablebit[4] = True
THEN HighAlarm := True ELSE HighAlarm := False;

SWNo $A8: Lowlevel.

LowLevel is a limit switch for the Power variable having the following function:

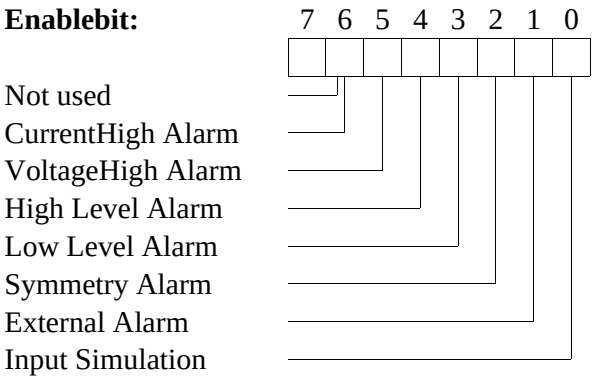
IF Power < LowLevel AND ChConfig.Enablebit[3] = True
THEN LowAlarm := True ELSE LowAlarm := False;

SWNo $A9: ChConfig.

The channel configuration for the Power Monitor channel is stored in a record having the following structure:

```
Record
    Enablebit: Array[0..7] of Boolean;   (* Offset = 0 *)
    Functions: BYTE;                     (* Offset = 1 *)
    Ref_A: BYTE;                         (* Offset = 2 *)
    Ref_B: BYTE;                         (* Offset = 3 *)
End;
```

The fields in ChConfig have the following interpretation:



CurrentHigh Alarm and VoltageHigh Alarm , are input overload alarms for the measured currents and voltages. If the alarm is enabled, the overload of a single input will set the corresponding alarm. The CurrentHigh Alarm and VoltageHigh Alarm are peak detecting. Depending on the signal waveform, an overload alarm may arise even though the measured RMS value is within the fullscale value.

HighLevel Alarm and LowLevel Alarm ,are user presettable limit switches for the Power variable.

Symmetry Alarm detects generator defects, by comparing the line currents. Unequal currents may be caused by a defective, short-circuited or disconnected phase in a generator.

External Alarm is a high level and a low level alarm for the LineFreq variable. See also the description for Bit1 in ChError.

If input simulation is enabled, the automatic updating of measured and calculated variables is disabled. Any value can be written to the variables for test purposes during commisioning. The alarms maintain their normal function, but use the simulated values. Automatic calculation of fullscale values is disabled.

Functions:
The functions field value is defined in a hexadecimal format, where the most significant digit is used to specify the channel functions.

Functions = $00 => Channel disabled.
Functions = $10 => Three-phase monitor (Three-phase system without neutral).
Functions = $20 => Single-phase monitor (Three single phases with neutral).

If the channel is not used, it should be disabled by writing "00" in ChConfig.Functions, otherwise errors may occur.

Ref_A and **Ref_B** are not used in the Power Monitor Channel.

SWNo $AA: LineFreq.
The LineFreq variable holds the frequency measured at the voltage input terminal marked U1.

SWNo $AB: FullScale.
FullScale is a record having the following structure:

```
Record
    Current: Real;      (* FullScale for current inputs *)      (* Offset = 0 *)
    VoltageU: Real;     (* FullScale for Voltage.U input *)     (* Offset = 4 *)
    VoltageV: Real;     (* FullScale for Voltage.V input *)     (* Offset = 8 *)
    VoltageW: Real;     (* FullScale for Voltage.W input*)      (* Offset = 12 *)
End;
```

FullScale.Current is a common fullscale for the current inputs. The current inputs are calibrated to be identical, so the chosen current mesurement transformers must be of the same type and ratio.

FullScale.VoltageU, FullScale.VoltageV and FullScale.VoltageW, are separate fullscale values for the voltage inputs. The FullScale values can be written directly to the variables or they can be calculated automatically. Automatic calculation is disabled when the channel is in input simulation mode.

Example: A 10:1 ratio current transformer is selected to match the current to be measured, with the nominal 1A RMS input current of the module.
The 10A RMS primary current needed to give 1A RMS at the input of the module, can be written directly into FullScale.Current. The value of FullScale.Current can also be calculated automatically, by writing the readout of a reference amperemeter, connected in series with the primary of the transformer in L1, into the Current.U variable. The calibration current used for automatic calculation of fullscale, should be larger than 20% of fullscale.

Example: A 230:24 ratio voltage transformer is selected to match the voltage to be measured, with the nominal 24V RMS input voltage of the module.
The 230V RMS primary voltage needed to give 24V RMS at the input of the module, can be written directly to the FullScale.VoltageU/V/W variables. The fullscale values for the voltage inputs can also be calculated automatically, by writing the readout of a reference voltmeter connected to the primary side of the measurement transformer, to the corresponding Voltage.U/V/W variables. The calibration voltage should be a reasonable fraction of fullscale.

SWNo $AC: Factors.
The factors variable is a record having the following structure:

```
Record
    PowerScale: Real;       (* Offset = 0 *)
    EnergyScale: Real;      (* Offset = 4 *)
End;
```

The Factors variables, PowerScale and EnergyScale, are unit conversion factors for the readout of power and energy.

Example: PowerScale and EnergyScale settings.

PowerScale	Power units	EnergyScale	Energy units
1	[W]	1	[Wh]
0.001	[kW]	0.001	[kWh]
0.86	[kcal]	0.86	[kcal]
3.412	[Btu/h]	3.412	[Btu]

SWNo $AD: Maintenance.

The Maintenance variable is a record having the following structure:

```
Record
    Date: Byte;       (* Offset = 0 *)
    Month: Byte;      (* Offset = 1 *)
    Year: Byte;       (* Offset = 2 *)
    Category: Byte;   (* Offset = 3 *)
End;
```

The Maintenance variable is used for service management and maintenance purposes, and holds the last date of service and indicates the type of service.

SWNo $AE: ChType.

For the Power Monitor channel, ChType has the following structure:

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

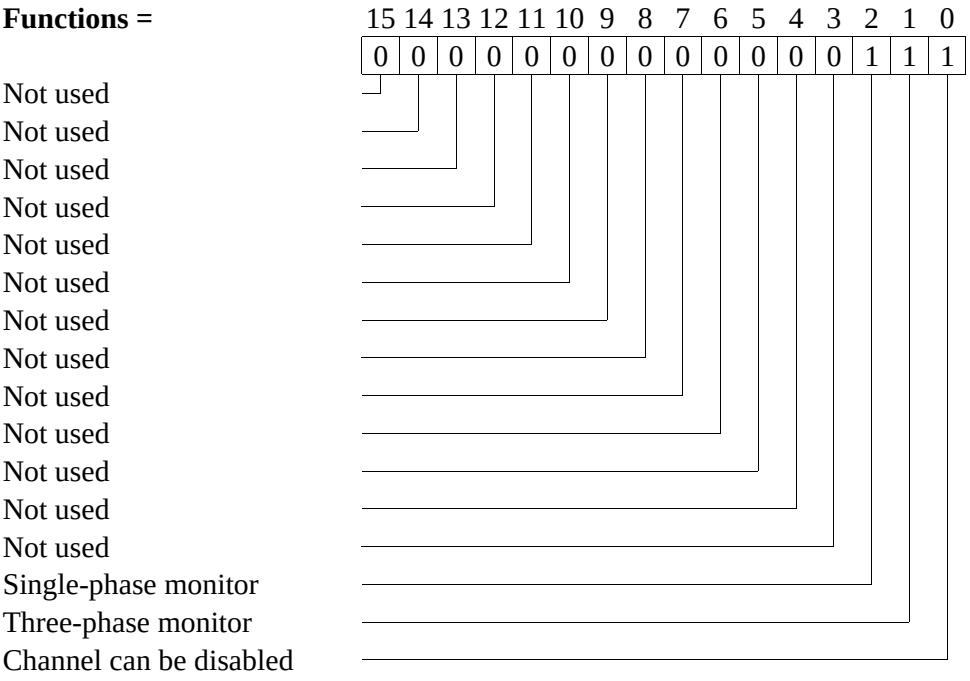
ChType has the following value:

ChannelType = 12

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	1	1	1	1	1	1	1	1

Functions =



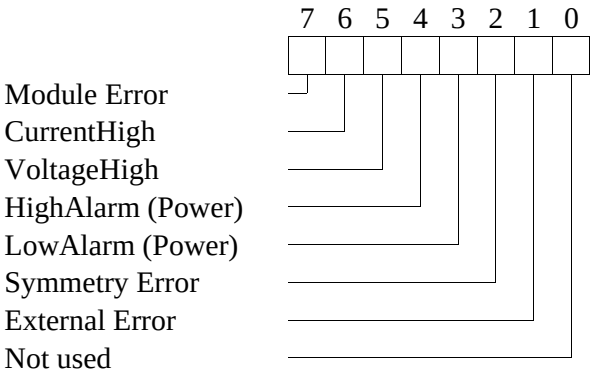
SWNo $AF: ChError.

ChError is a record of the following structure:

```
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.

ChError:



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

Bit 6 CurrentHigh is set if the **peak current** at one of the input terminals U2, V2 or W2 exceeds the max. RMS value (1A) by more than approx. 100 % and ChConfig.-EnableBit[6] = True.

Bit 5 VoltageHigh is set if the **peak voltage** at one of the input terminals U1, V1 or W1 exceeds the max. RMS value (24V) by more than approx. 100 % and ChConfig.-EnableBit[5] = True.

Bit 4 HighAlarm is set if Power > HighLevel and ChConfig.Enablebit[4] = True.

Bit 3 LowAlarm is set if Power < LowLevel and ChConfig.Enablebit[3] = True.

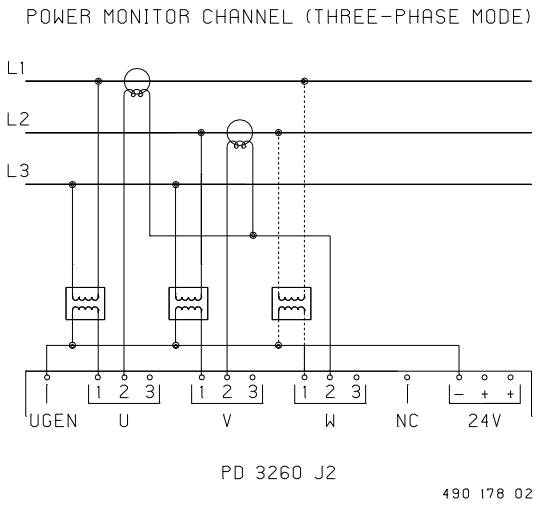
Bit 2 Symmetry Error is set if the sum of currents is larger than 40% of the value in FullScale.Current, and one of the currents differs from 1/3 of the sum of currents by more than 20%, during 5 periods of the line frequency and ChConfig.Enablebit[2] = True.

Bit 1 External Error is set if the measured LineFreq at U1 is outside the range 45 to 62 Hz. This would occur if the voltage at Ur has become disconnected or is outside the frequency range.

5.1 Connections to the Power Monitor Channel.

The Power Monitor Channel has two basic connection schemes, depending on the channel configuration. One is for three-phase systems and one is for single-phase systems.

Three-phase system without neutral (ChConfig.Functions = $10).
In a three-phase system, the total power of a three-phase power consumer can be measured as the sum of the readouts of three power meters, each measuring a phase voltage and a line current. If the common reference for the phase voltage measurement is connected to one of the lines, the phase voltage measured on this line will be zero, and the corresponding power meter will also read zero and can be disregarded. The voltages measured at the remaining two power meters are now the line voltages.



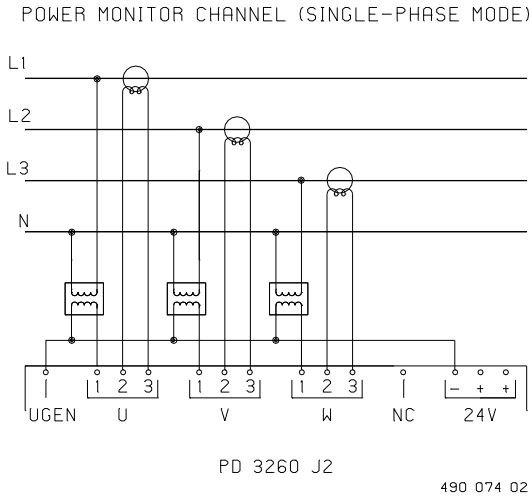
When the power monitor channel is in three-phase mode, only two voltage measurement transformers are required to measure the L1 and L2 line voltages, referenced to the L3 line, which is used as the common for the primary side of the voltage transformers. The secondary side of the transformers are all referenced to the 24V- terminal on the module.
The line voltage between the L1 and L2 lines may be measured by a third voltage transformer if required. It is recommended that any unused voltage inputs are connected to the 24V- terminal.

Voltage measurement transformers must be chosen to match the **line voltages** with the voltage inputs of the module. The voltage inputs are high precision factory calibrated.

The line currents are measured with two current transformers in the L1 and L2 lines. One side of each secondary of the current transformers must be connected to the corresponding positive current input terminals U2 and V2 on the module. The other side is commoned at the positive current input terminal W2. In this way, all three line currents are measured using only two current measurement transformers.

The current inputs share a common FullScale.Current, so identical current transformers must be chosen to match the **line currents** with the current inputs of the module. The current inputs are high precision factory calibrated.

Three-phase system with neutral (ChConfig.Functions = $20):



In a single-phase mode, the module may be used to monitor up to three power consuming devices on the same, or different phases, by using the current and voltage inputs separately.

The primary side of the voltage transformers used to measure the phase voltages are referenced to neutral.
The secondary sides are referenced to the 24V- terminal on the module. It is recommended that unused voltage inputs are connected to the 24V- terminal. The U1 voltage input must always be used, because the frequency measurement is performed using this input.

The voltage transformers must be chosen to match the **phase voltages** with the voltage inputs of the module. The voltage inputs are high precision factory calibrated.

The line currents are measured with current transformers connected to each line, and the secondary side is connected to the corresponding current inputs on the module.

The current transformers must be chosen to match the **line currents** with the current inputs of the module. Identical current transformers must be used, since the inputs share a common FullScale.Current factor. The current inputs are high precision factory calibrated.

6 Generator Switch Channel (channel B).

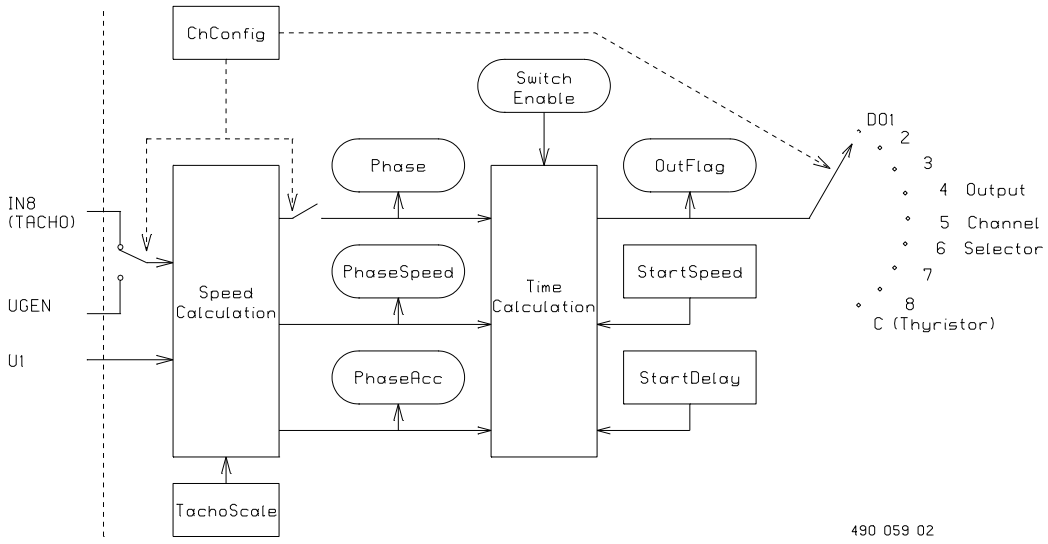
The Generator switch channel can be used, when an asynchronous or synchronous generator is to be coupled to a power line.

Asynchronous generator:
The generator speed and acceleration is measured with the tachometer input to calculate when the generator speed will match the line frequency. The generator is connected at this moment in time, if the difference between generator and line frequency is less than a configured value.

Synchronous generator:
The difference in phase, phase speed, and phase acceleration between the line voltage and generator voltage is measured to calculate when the phase shift will be zero. The generator is connected at this moment in time, if the difference between generator and line frequency is less than a configured value.

The Generator Switch Channel can control a digital output channel or the Thyristor Switch Channel. It is also possible for the coupling instance to be brought forward by a known delay in the switching element.

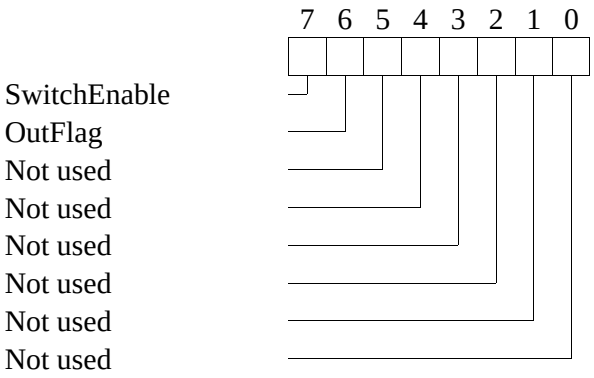
Block diagram of generator switch function:



Variables on Generator Switch channel. Channel identifier: **GeneratorSwitch**

SWNo	Identifier	Memory type	Read out	Type	SI Unit
B0	FlagReg	RAM Read Write	Binary	Array	- - -
B1	Phase	RAM Read Write	Decimal	Real	°
B2	PhaseSpeed	RAM Read Write	Decimal	Real	°/s
B3	PhaseAcc	RAM Read Write	Decimal	Real	°/s/s
B4					
B5					
B6					
B7					
B8					
B9	ChConfig	EEPROM RPW	- - -	Record	- - -
BA	TachoScale	EEPROM RPW	Decimal	Real	[]
BB	StartSpeed	EEPROM RPW	Decimal	Real	°/s
BC	DelayComp	EEPROM RPW	Decimal	Real	s
BD	Maintenance	EEPROM RPW	- - -	Record	- - -
BE	ChType	PROM Read Only	- - -	Record	- - -
BF	ChError	RAM Read Only	Binary	Record	- - -

SWNo $B0: FlagReg.



Bit 7: SwitchEnable.
The start function is enabled by setting SwitchEnable = True. When the switching conditions are satisfied, the OutFlag is set, and the controlled output channel is switched on. The Outflag is reset and the controlled output is switched off, by setting SwitchEnable = False.

Bit 6: OutFlag.
OutFlag indicates the actual status of the output function. If a digital I/O channel is selected in Ref_A, OutFlag controls the OutFlag of this digital I/O channel.

SWNo $B1: Phase.
The Phase variable is only used when a synchronous generator is configured. It holds the phase shift between the measured line voltage and the measured synchronous generator voltage, in degrees. The line voltage is measured with a voltage transformer with the primary connected between the L1 and L3 lines, and the generator voltage is measured with another voltage transformer also connected at the L1 and L3 line, but on the other side of the switching element. See diagram in chapter 6.1.

SWNo $B2: PhaseSpeed.
The PhaseSpeed variable holds the phase speed between the generator and line frequency.

Calculation: PhaseSpeed := (line frequency - generator frequency) * 360° [°/s]

SWNo $B3: PhaseAcc.
The PhaseAcc variable holds the phase acceleration between the generator and line frequency.

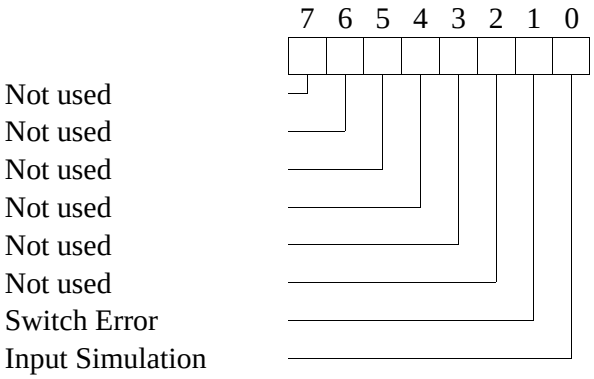
Calculation: PhaseAcc := (PhaseSpeed - OldPhaseSpeed) / DeltaT [°/s/s]

SWNo $B9: ChConfig.
The channel configuration for the Generator Switch channel is stored in a record having the following structure:

```
Record
    Enablebit: Array[0..7] of Boolean;   (* Offset = 0 *)
    Functions: BYTE;                     (* Offset = 1 *)
    Ref_A: BYTE;                         (* Offset = 2 *)
    Ref_B: BYTE;                         (* Offset = 3 *)
End;
```

The fields in ChConfig have the following interpretation:

Enablebit:



During an attempt to connect a generator to a power line, it may be that the module fails to switch, because zero phase shift between line and generator does not occur within the phase speed difference range, as defined in the StartSpeed variable. A Switch Error indicates that switching has failed, and the generator frequency has accelerated beyond the line frequency by the frequency difference specified in StartSpeed. Detection of Switch Error is enabled by setting ChConfig.EnableBit[1] to true.

In input simulation mode, no variables in the channel are automatically calculated and no variable values will be overwritten by the module. This makes it possible to simulate any value in a variable for test purposes. Input simulation is enabled by setting ChConfig.EnableBit[0] to true.

Functions:
The functions field is used in a hexadecimal format, where the most significant digit is used to specify the channel functions.

Functions = $00 => Channel disabled.
Functions = $10 => Asynchronous generator switch.
Functions = $20 => Synchronous generator switch.

If the channel is not used, the channel should be disabled by writing "00" in ChConfig.Functions. Otherwise errors may occur.

Asynchronous generator switch (Functions = $10)
The generator speed and acceleration is measured with a tachometer connected to the DI 8 terminal. Digital I/O 8 must be configured as an input: Functions = $00. The moment when the generator will achieve the configured StartSpeed is calculated and compensated by the configured DelayComp.
The switch function is enabled by setting FlagReg.SwitchEnable = True. The output selected in Ref_A is switched on at the calculated instant, and is switched off by setting FlagReg.SwitchEnable= False.

Precise function description:

After reset:
SwitchEnable := False;
OutFlag := False;

Loop
IF (SwitchEnable = True) AND (ABS(PhaseSpeed) <= StartSpeed)
THEN OutFlag := True at calculated time when PhaseSpeed = 0;
IF SwitchEnable = False THEN OutFlag := False;
End

Synchronous generator switch (Functions = $20)
The generator speed and acceleration is measured via the generator voltage input UrGEN.
When the generator has obtained the configured StartSpeed, the moment in time when line and generator voltages will be in phase is calculated and compensated by the configured relay Start-Delay.
The switch function is enabled by setting SwitchEnable = True. The output selected in Ref_A is switched on at the calculated instant, and is switched off by setting SwitchEnable = False.

Precise function description:

After reset:
SwitchEnable := False;
OutFlag := False;

Loop
IF (SwitchEnable = True) AND (ABS(PhaseSpeed) <= StartSpeed)
THEN OutFlag := True at calculated time when Phase = 0;
IF SwitchEnable = False THEN OutFlag := False;
End

Ref_A:
Ref_A holds the number of the channel to be controlled. If a Digital I/O Channel is selected, it must be configured as an output (Functions = $10). If the Thyristor Switch Channel is selected, it must be enabled, and two or more of the digital I/O´s must be configured to be controlled by the Thyristor Switch channel (Functions = $40 for the digital output channels).

SWNo $BA: TachoScale.
The TachoScale variable is only used when an asynchronous generator is selected in the Functions configuration.
The scale factor that converts the tachometer frequency to generator voltage frequency is configured in the TachoScale variable. For best results, choose a tachometer giving a pulse frequency equal to the generator frequency at nominal generator speed i.e. TachoScale = 1.

Calculation:
TachoScale := frequency * 60 / generator rotations per minute / tachometer pulses per revolution
Example: TachoScale := 50 Hz * 60 / 1500 RPM / 2 PPR = 1.0

SWNo $BB: StartSpeed.
The maximum phase speed difference between line and generator frequency within which the generator can be coupled, must be inserted in the StartSpeed variable. StartSpeed is a tolerence parameter, and in the example below, the frequency range is to be 48 Hz to 52 Hz.

Calculation: StartSpeed := frequency difference * 360°
Example: StartSpeed := (50 Hz - 48 Hz) * 360° = 720 [°/s]

StartSpeed is typically set to a small value when configured for an asynchronous generator.
For synchronous generators, some frequency difference must be allowed, otherwise the module may not be able to couple the generator, because a zero phase shift between line and generator does not occur within the specified phase speed difference range.

SWNo $BC: DelayComp.
The calculated start time for generator coupling, can be modified by inserting a value in the DelayComp variable, to compensate for relay contact delay or to allow magnetisation build up in the generator. A positive value, in seconds, will force the generator to be switched before the calculated start time.

SWNo BD: Maintenance.
The Maintenance variable is a record of the following structure:

```
Record
    Date: Byte;       (* Offset = 1 *)
    Month: Byte;      (* Offset = 2 *)
    Year: Byte;       (* Offset = 3 *)
    Category: Byte;   (* Offset = 4 *)
End;
```

The Maintenance variable is used for service management and maintenance purposes, and holds the last date of service and indicates the type of service.

SWNo $BE: ChType.
For the Generator Switch Channel ChType is of the following structure:

```
Record
    ChannelType: Word;                  (* Offset = 0 *)
    Exist: Array[0..15] of Boolean;     (* Offset = 2 *)
    Functions: Array[0..15] of Boolean; (* Offset = 4 *)
End;
```

The Generator Switch Channel is a company specific channel, where ChType has the following value:

ChannelType = $8008

Exist =

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

Functions =

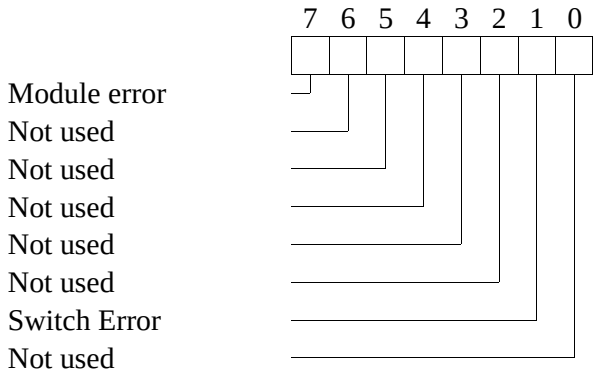
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Synchronous generator
Asynchronous generator
Channel can be disabled

SWNo $BF: ChError.
ChError is a record of the following structure:

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

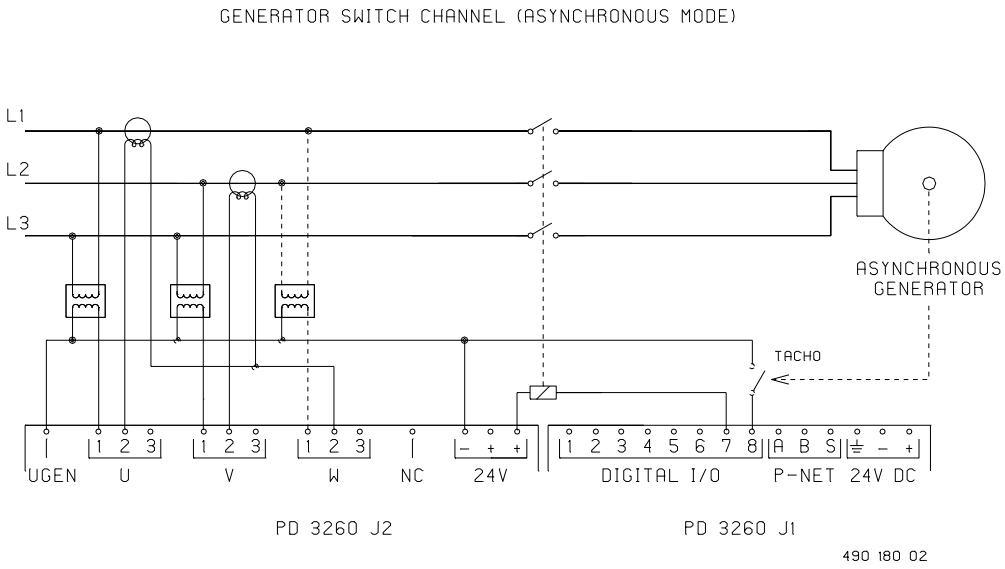
The 8 bits in ChError.His and ChError.Act have the following meaning. When an error occurs, the corresponding bit is set in both ChError.His and ChError.Act. When the error disappears, the bit is cleared in ChError.Act.



Bit 7 Module Error: If this bit is set, the rest of the bits have no meaning, because a Module error can lead to random error codes on individual channels (also see "Service channel").

Bit 1 Switch Error: Generator not connected, due to the fact that the generator frequency has passed the line frequency by the frequency difference specified in StartSpeed. The value inserted in StartSpeed may be too small.

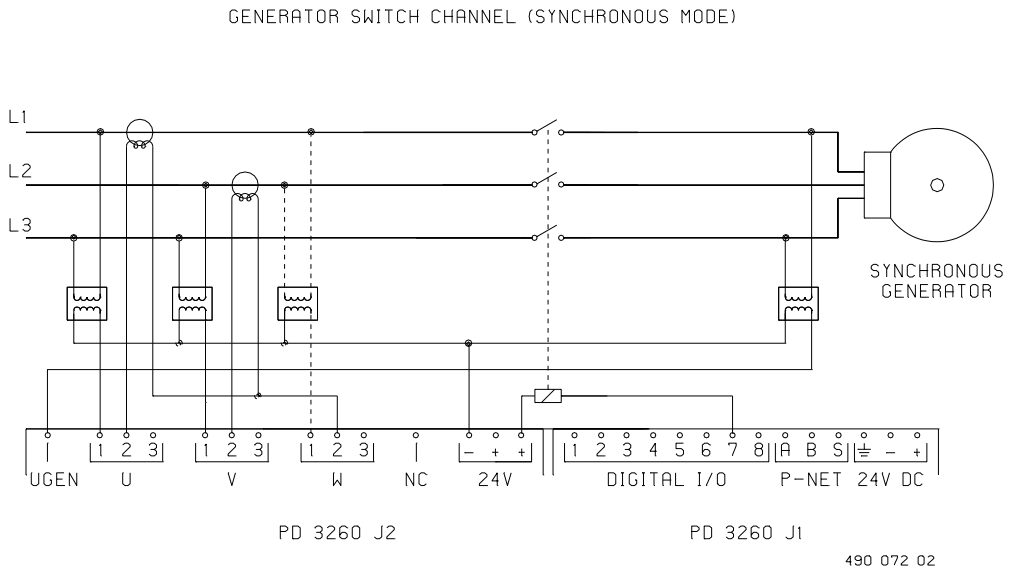
6.1 Connections to Generator Switch Channel.



Asynchronous generator connection:
The diagram shows a system where the Power Monitor Channel and the Generator Switch Channel are used for controlling a digital output channel to drive a relay. For systems with thyristor output, see chapter 7.1: Connections to Thyristor Switch Channel.

When using an asynchronous generator, the generator speed is measured with a tachometer connected to the digital I/O 8 terminal. Any of the remaining 7 I/O's may be configured as the controlling output from the Generator Switch Channel. Using I/O 7 as the output is the best choice, since I/O 1 to I/O 6 can also be used by the Thyristor Switch Channel to perform thyristor triggering.

Only the voltage transformer between L1 and L2 and the tachometer input are necessary inputs for the asynchronous generator switch function. Additional current and voltage transformers may optionally be connected, if Power Monitor Channel functions are required. It is recommended that unused voltage inputs are connected to the 24V- terminal, otherwise noise at the inputs may generate errors.



Synchronous generator connection:
The diagram shows a system where the Power Monitor Channel and the Generator Switch Channel are used for controlling a digital output channel to drive a relay. For systems with thyristor output, see chapter 7.1: Connections to Thyristor Switch Channel.

When using a synchronous generator, the generator speed is measured using a voltage transformer connected to the generator terminal UGEN. The primary of the transformer must be connected between the L1 and L3 lines in a similar way to the line voltage transformer between L1 and L3, but on the other side of the switching element.

Only the voltage transformers connected between L1 and L3 on the power line and at the generator, are nessecary inputs for the synchronous generator switch function. Additional current and voltage transformers may be optionally connected, if Power Monitor Channel functions are required. It is recommended that unused voltage inputs are connected to the 24V- terminal, otherwise noise at the inputs may generate errors.

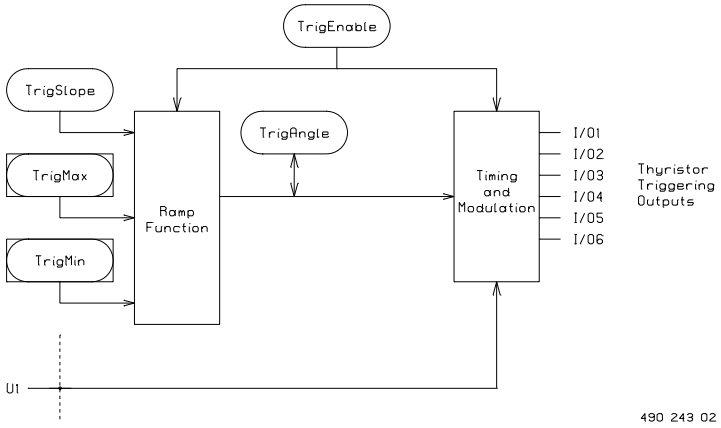
7 Thyristor Switch Channel (channel C).

The Thyristor Switch Channel performs ramp controlled thyristor triggering for a one to three-phase thyristor switch. Triggering is referred to zero voltage crossing at the U1 voltage input. The digital outputs I/O 1 to I/O 6 can be configured to be the controlling outputs for the Thyristor Switch Channel.

Variables on Thyristor Switch channel. Channel identifier: **ThyristorSwitch**

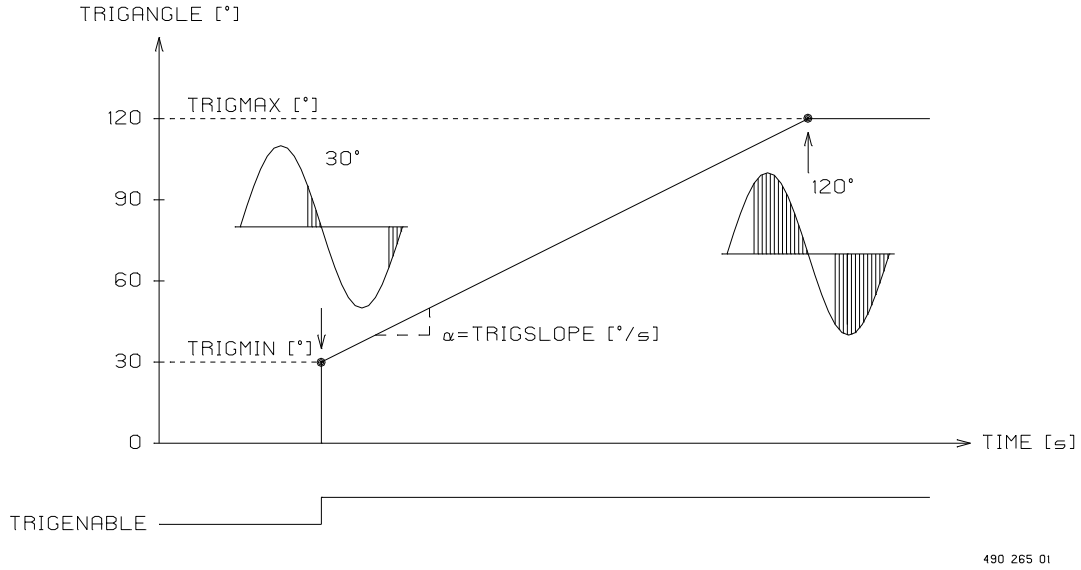
SWNo	Identifier	Memory type	Read out	Type	SI Unit
C0	FlagReg	RAM Read Write	Binary	Array	- - -
C1	TrigAngle	RAM Read Write	Decimal	Real	°
C2					
C3					
C4					
C5					
C6	TrigSlope	RAM Init EEPROM	Decimal	Real	°/s
C7					
C8					
C9	ChConfig	EEPROM RPW	- - - - - -	Record	- - -
CA					
CB	TrigMax	RAM Init EEPROM	Decimal	Real	°
CC	TrigMin	RAM Init EEPROM	Decimal	Real	°
CD	Maintenance	EEPROM RPW	- - - - - -	Record	- - -
CE	ChType	PROM Read Only	- - - - - -	Record	- - -
CF	ChError	RAM Read Only	Binary	Record	- - -

Block diagram of thyristor switch function:

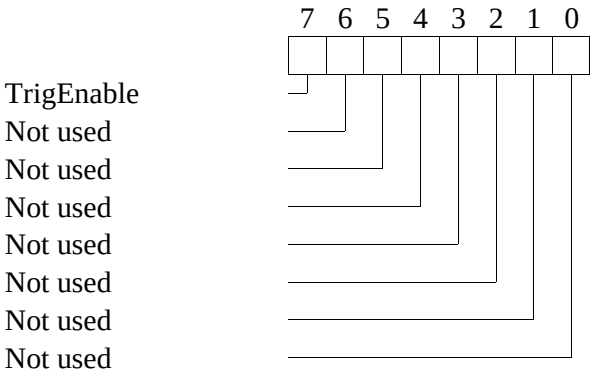


The ramp function in the Thyristor Switch Channel controls the thyristor coupling angle with respect to the zero voltage crossings measured at U1.

It is a simple, user specified up/down ramp, with slope (change of coupling angle per second), that is configured in the TrigSlope variable. Lower and upper limits for the coupling angle are configured in the TrigMin and TrigMax variables. See also the diagram below. The thyristor coupling angle is the output from the ramp, and can be read in the TrigAngle variable.
The ramp function is enabled or disabled with the TrigEnable bit in the FlagReg variable. Up or down ramping is controlled by the sign of the TrigSlope variable.



SWNo $C0: FlagReg.



Bit 7: TrigEnable.
The TrigEnable bit controls the trigger output function. If TrigEnable = True, the ramp function is performed, and if TrigEnable = False, all thyristors are switched off and the ramp function is stopped. After reset, TrigEnable is set to false.

SWNo $C1: TrigAngle.
The TrigAngle variable is the output from the ramp function, and holds the actual required thyristor coupling degree. The variable is ramped in time by the slope variable, and stops at the limits TrigMax and TrigMin.

Precise function description of ramp function:

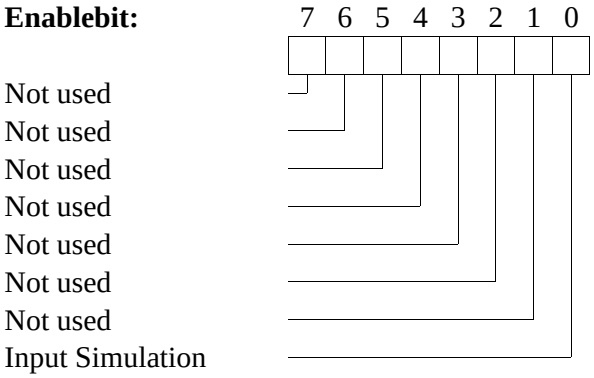
```
Loop
    IF TrigEnable = True THEN
    BEGIN
        TrigAngle := TrigAngle + TrigSlope/LineFreq;
        IF TrigAngle > TrigMax THEN TrigAngle := TrigMax;
        IF TrigAngle < TrigMin THEN TrigAngle := TrigMin;
    END
    ELSE Thyristors := Off;
END;  (* Loop *)
```

SWNo $C6: TrigSlope.
TrigSlope holds the slope of the thyristor coupling ramp. The ramp slope is measured in deg/s. Up ramping is obtained by writing a positive value in the TrigSlope variable.

SWNo $C9: ChConfig.
The channel configuration for the Thyristor Switch Channel is stored in a record having the following structure:

```
Record
    Enablebit: Array[0..7] of Boolean;   (* Offset = 0 *)
    Functions: BYTE;                     (* Offset = 1 *)
    Ref_A: BYTE;      Not used           (* Offset = 2 *)
    Ref_B: BYTE;      Not used           (* Offset = 3 *)
End;
```

The fields in ChConfig have the following interpretation:



If input simulation is enabled, automatic updating of measured and calculated variables is disabled. Any value can be written to the variables, for test purposes during commisioning.

Functions:
Functions = $00 => Channel disabled.
Functions = $10 => Ramp controlled thyristor switch (zero voltage crossing).
If the channel is not used, the channel should be disabled by writing "00" in ChConfig.Functions. Otherwise errors may occur.

Functions = $10: Ramp controlled thyristor switch (zero voltage crossing).
Ramp controlled switching with thyristor coupling angle related to zero voltage crossing.

The digital outputs I/O 1 to I/O 6 can be configured to be the controlling outputs for the Thyristor Switch Channel, by setting their individual ChConfig.Functions to $40. The Thyristor Switch Channel uses the digital outputs in pairs: I/O 1 & I/O 2, I/O 3 & I/O 4 and I/O 5 & I/O 6. Both of the I/O´s in a pair must be configured correctly, otherwise none of them will be used. The digital I/O 7 and I/O 8 cannot be used by the Thyristor Switch Channel.

The output from a digital output, controlled by the Thyristor Switch Channel, is a modulated 10kHz signal. The reading of flags in the I/O channels, at this frequency, has no meaning.

SWNo $CB: TrigMax.
The maximum thyristor trigger angle (0 - 180 degrees) for the thyristor ramp function, must be inserted in the TrigMax variable.

SWNo $CC: TrigMin.
The minimum thyristor trigger angle (0 - 180 degrees) for the thyristor ramp function, must be inserted in the TrigMin variable.

SWNo $CD: Maintenance.
The Maintenance variable is a record of the following structure:

```
Record
    Date: Byte;       (* Offset = 0 *)
    Month: Byte;      (* Offset = 1 *)
    Year: Byte;       (* Offset = 2 *)
    Category: Byte;   (* Offset = 3 *)
End;
```

The Maintenance variable is used for service management and maintenance purposes, and holds the last date of service, and indicates the type of service.

SWNo $CE: ChType.
For the Thyristor Switch Channel, ChType has the following structure:

```
Record
    ChannelType: Word;              (* Offset = 0 *)
    Exist: Array[0..15] of Boolean; (* Offset = 2 *)
    Functions: [0..15] of Boolean;  (* Offset = 4 *)
End;
```

The Generator Switch Channel is a company specific channel, where ChType has the following value:

ChannelType = $8003

Exist=

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

Functions =

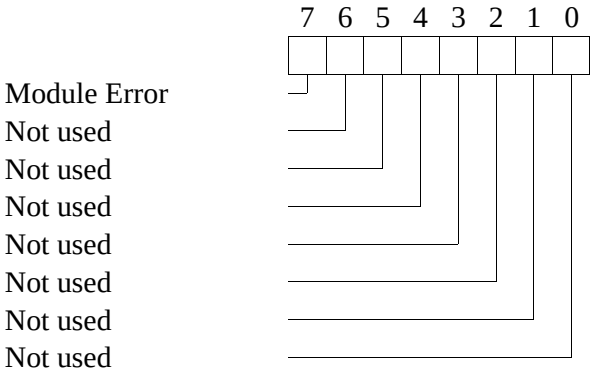
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Not used
Voltage zero crossing
Channel can be disabled

SWNo $CF: ChError.
ChError is a record of the following structure:

```
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.

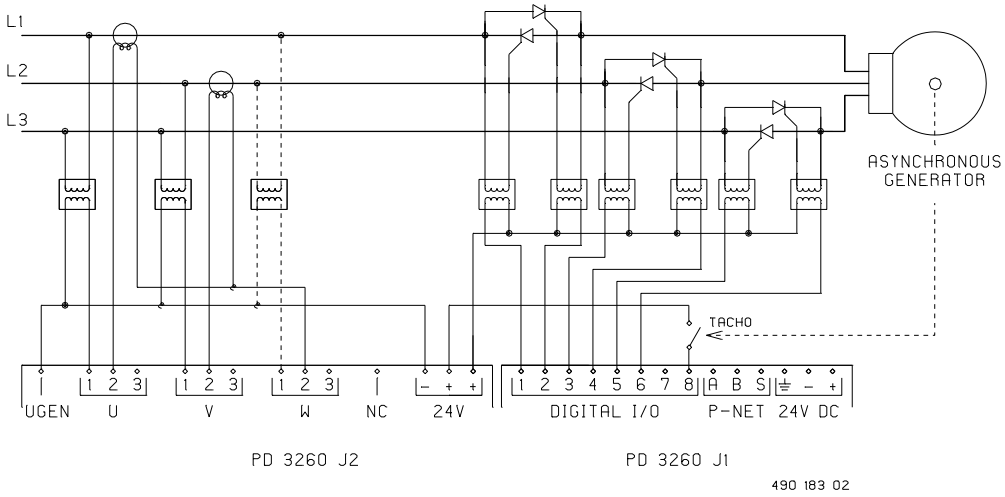


Bit 7 Module error: A module error can lead to random error codes on individual channels. See the "Service channel" for further information on module errors.

7.1 Connections to Thyristor Switch Channel.

The Thyristor Switch Channel has six thyristor triggering outputs, to control a three-phase switch having six thyristors. The IO-channels are used in pairs, where both channels in the pair must be configured to be controlled by the Thyristor Switch Channel (Functions = $40).

The diagram below shows a system with an asyncronous generator, which utilises the Power Monitor Channel, the Generator Switch Channel and the Thyristor Switch Channel for driving a three-phase thyristor switch.



The Thyristor Switch Channel always uses the voltage at the U1 input as a reference for triggering.

The six transformers on the digital outputs are used to to pass the 10kHz modulated triggering signals to the thyristors. For pulse transformer selection and application, refer to relevant literature and data sheets for the selected thyristors.

The additional voltage and current inputs shown in the diagram, are used by the Power Monitor Channel, and are not necessary for the thyristor switch function. Also see the description of the Power Monitor Channel. The tachometer connected to digital I/O 8, is used by the Generator Switch Channel, to measure the speed and acceleration of the asynchronous generator. If a synchronous generator is used, the tachometer connection is replaced by the analog voltage input on UGEN. See also the description of the Generator Switch Channel.

8 Calculator program.

The PD 3260 is able to perform arithmetical and boolean functions by means of a calculator program. All variables within the module can be used within the expressions.

The calculator has a real type accumulator, a boolean type accumulator, two channel pointers, two index registers and a bit index register. The instruction set includes Move, Compare, Jump, logical operations and arithmetical operations. Some of the instructions can operate either on variables via SWNo (channel no. and register no.) or on immediate values.

It is not possible to write directly into EEPROM by means of the Calculator program. However, it is possible to save all variables with memory type **RAM AutoSave** to EEPROM, directly from the Calculator program. The variables in the DataChannel may be used for common configuration parameters and temporary values.

The calculator program is stored (download and upload) and controlled (start, stop etc.) using the Program Channel (channel $D). The program may be downloaded from a PC, eg. via the Calculator Assembler program, or from a master controller unit within the P-NET system, eg. PD 5000 with a Process-Pascal program installed.

The execution time for each instruction is approximately 0.4 ms. By disabling the automatic functions in the channels which are not in use, the operating speed of the calculator program may be slightly increased.

Some typical instructions, with execution times, are shown below. The conditions for the time measurements are: All the channels in the module are enabled, the digital outputs 1-6 are configured to be controlled by the Thyristor Switch Channel. The minimum time includes occasional P-NET communication, whereas the maximum time includes significant P-NET communication.

Instruction	**Min time**	**Max time**
Move CR2:0, Acc	0.30 ms	0.32 ms
Move Acc, CR2:7[IR2]	0.41 ms	0.43 ms
Move #0E, CR2	0.18 ms	0.20 ms
Div 123.4	0.46 ms	0.48 ms
Mul 123.4	0.43 ms	0.45 ms
Sub 123.4	0.37 ms	0.40 ms
Add 123.4	0.36 ms	0.38 ms
LookUp CR2:#C (each index pair)	2.38 ms	2.51 ms

Refer to the PD Calculator Assembler Manual, PD no. **502 061**, for a list of the available instructions and information on how to program the calculator.

9 Program Channel (channel D).

The Program Channel utilises Program Invocation Management, and specifies some general information about a Calculator program in the PD 3260. The Calculator program is an application program, which can be downloaded or uploaded via this channel, and the status of the program can be supervised and changed.

The program in the library is indexed by a number, LibraryIndex, which is found in Library-Control. The total number of programmes, which can be stored in the library memory is stated in MaxLibraryIndex, which is found in LibraryStatus. The total number of programmes that can be held within the PD 3260 module is 1.

The selected program can be controlled and supervised. The commands and program states conform to the instructions specified in Manufacturing Message Specification (MMS).

Registers on Program Channel (channel n). Channel identifier: **Calculator**

SWNo	Identifier	Memory Type	Read Out	Type
D0	ProgramControl	RAM Read Write	- - - - - - -	Record
D1	ProgramStatus	RAM Read Only	- - - - - - -	Record
D2	ProgramID	Read Only	- - - - - - -	Record
D5	SystemPointer	Read Only	Hex	LongInteger
D7	MemoryInfo	Read Only	- - - - - - -	Record
D8	IDAndCode	Special function	- - - - - - -	Record
D9	ChConfig	EEPROM RPW	- - - - - - -	Record
DA	LibraryControl	RAM Read Write	- - - - - - -	Record
DB	LibraryStatus	RAM Read Only	- - - - - - -	Record
DC	LibraryProgramID	Read Only	- - - - - - -	Record
DD	Maintenance	EEPROM RPW	- - - - - - -	Record
DE	ChType	PROM Read Only	- - - - - - -	Record
DF	ChError	RAM Read Only	- - - - - - -	Record

SWNo $D0: ProgramControl

ProgramControl is used to set and change the state of the current program, which has been selected and invoked via the Program Channel. The selected program number is inserted and shown as a part of ProgramControl. Commands can be sent to ProgramControl, to stop/start or reset the program.

ProgramControl is a record of the following structure:

```
Record
    Command         : BYTE;     (* Offset = 0 *)
    ProgramToSelect : Word;     (* Offset = 2 *)
    ErrorStatus     : Bit32;    (* Offset = 4 *)
End
```

Command is used to send a command to the Program Channel to change the state of the current program. A list of possible commands is given below. The commands and the corresponding numbers conform to the Request Instructions used by MMS.

Command		**Purpose**
38	SelectProgram	Selects a program from the library, resulting in ProgramStatus.State = Idle, if the program is OK. (MMS = CreateProgramInvocation)
39	UnSelectProgram	Sets SelectedProgram to 0, resulting in ProgramStatus.State = Non-selected. (MMS =DeleteProgramInvocation)
40	Start	Starts the selected program. (MMS=Start)
41	Stop	Stops the selected program. (MMS=Stop)
42	Resume	Continues program execution of the selected program. (MMS=Resume)
43	Reset	Resets the selected program. (MMS=Reset)
44	Kill	Stops the selected program instantly and sets the ProgramStatus.State = Unrunable. (MMS=Kill)

Command is automatically set to 0 by the Program Channel after writing to the variable. ProgramStatus.State is immediately updated to one of the corresponding temporary states: Starting, Stopping, Resuming or Resetting, each time a command is sent to the Command variable. When the state is changed, it is possible to read the ProgramStatus.State variable, to check whether the Command was executed successfully or that the operation failed. If an error occurs following a Start or Resume command, the specific error may be read in ChError.

ProgramToSelect holds the library index for the program to select, or the already selected program. ProgramToSelect is copied from ChConfig following a module reset or power up.

ErrorStatus does not indicate any errors in the application program, and consequently all bits always read FALSE.

SWNo $D1: ProgramStatus

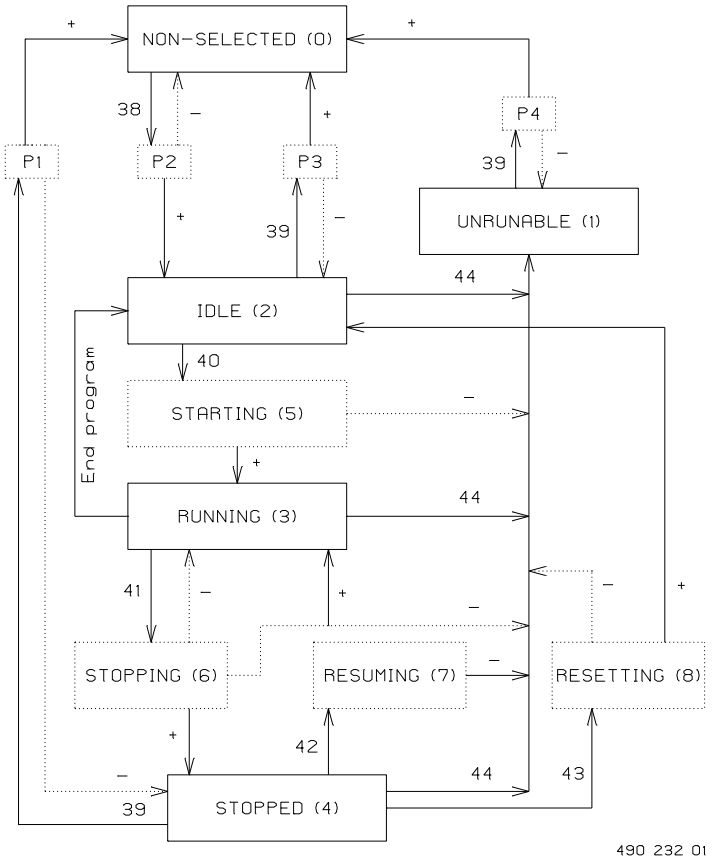
ProgramStatus summarises the state and error conditions for the selected program. The library index for the selected program is also indicated.

ProgramStatus is a record of the following structure:

```
Record
    State           : BYTE;     (* Offset = 0 *)
    SelectedProgram : Word;     (* Offset = 2 *)
    ErrorStatus     : Bit32;    (* Offset = 4 *)
End
```

State indicates the current state for the selected program, eg. stopping, stopped, running, idle, non-selected etc. A state diagram and a list of possible states is given below. The states and the corresponding numbers conform to the states for Program Invocation Management used by MMS.

PROGRAM INVOCATION STATE DIAGRAM



\+ State transition succeeded - State transition failed.

State		Explanation
0	Non-selected	No program selected. (MMS = Non-selected)
1	Unrunable	The program cannot run. (MMS = Unrunable)
2	Idle	The program is stopped and reset. (MMS = Idle)
3	Running	The program is running. (MMS = Running)
4	Stopped	The program is stopped. (MMS = Stopped)
5	Starting	The program is changing state from idle to running. (MMS = Starting)
6	Stopping	The program is changing state from running to stopping. (MMS = Stopping)
7	Resuming	The program is changing state from stopped to running. (MMS = Resuming)
8	Resetting	The program is changing state from stopped to idle. (MMS = Resetting)

SelectedProgram holds the library index for the selected program. SelectedProgram is 0, if State is Non-selected.
ErrorStatus is identical to ProgramControl.ErrorStatus, but only read access is possible. All bits in ErrorStatus always read FALSE.

SWNo $D2: ProgramID
ProgramID is used to identify the selected program. The record includes a name for the program, version number, the required version number for the interpreter program and a name of the software house that created the program. Compile time, compiler version and actual size for the program is also a part of ProgramID, as well as SumCheck (2's complement word addition without carry), and a code type identifier. The SumCheck value is used for check sum calculations when the program is started, i.e. following a Start command (40) or a Resume command (42). CodeType must match CodeType found in ChType. The CodeType check is performed at the same time as the SumCheck.

ProgramID is a record of the following structure:

```
Record
    ProgramName     : STRING[20];     (* Offset = 0 *)
    Version         : Word;           (* Offset = 22 *)
    InterpreterVers : Word;           (* Offset = 24 *)
    SoftwareHouse   : STRING[20];     (* Offset = 26 *)
    CompileTime     : DateTimeRec;    (* Offset = 48 *)
    CompilerVersion : Word;           (* Offset = 56 *)
    ActualSize      : LongInteger;    (* Offset = 58 *)
    SumCheck        : Word;           (* Offset = 62 *)
    CodeType        : Word;           (* Offset = 64 *)
    NoOfTask        : Word;           (* Offset = 66 *)
End
```

CodeType is 2, corresponding to calculator program code.
NoOfTask is always 1.

SWNo $D5: SystemPointer
This variable holds a pointer to system specific data. These system data may be used for debugging, read out of kernel data etc. Only PROCES-DATA knows the meaning of the information referred to in the SystemPointer.

SWNo $D7: MemoryInfo
For the program segment stored in the library, selected by LibraryControl.LibraryIndex, a corresponding memory information can be read. The memory information provides the actual size of the program, the max. size for the program segment, and a code for the memory type in which the program is stored.

MemoryInfo is a record of the following structure:

```
MemoryInfo : Record
    ActualSize      : LongInteger;    (* Offset = 0 *)
    MaxSize         : LongInteger;    (* Offset = 4 *)
    MemoryType      : Word;           (* Offset =8 *)
End
```

ActualSize includes the size of the program code and the header with the ProgramID.

MaxSize indicates the max size for the program segment, and is the max. size for a program to download within the memory area specified by **MemoryType**.
MaxSize = 7000 and **MemoryType = 4** in PD 3260.

SWNo $D8: IDAndCode
This SoftWire number is used for uploading or downloading programmes from/to the Program Channel. When a program is downloaded to the Program Channel, the entire program and a header with the ProgramID and size indicator, is stored in IDAndCode, by means of a LongStore instruction.

The program and the header with the ProgramID, a size indicator and a code-type indicator, is a variable of the following type:

```
IDAndCode = ARRAY[1..ActualSize] OF BYTE;
```

The format of the data stored in the first part of IDAndCode, exactly matches the format of ProgramID. IDAndCode can be interpreted as a record having the following structure:

```
Record
    ProgramName     : STRING[20];     (* Offset = 0 *)
    Version         : Word;           (* Offset = 22 *)
    InterpreterVers : Word;           (* Offset = 24 *)
    SoftwareHouse   : STRING[20];     (* Offset = 26 *)
    CompileTime     : DateTimeRec;    (* Offset = 48 *)
    CompilerVersion : Word;           (* Offset = 56 *)
    ActualSize      : LongInteger;    (* Offset = 58 *)
    SumCheck        : Word;           (* Offset = 62 *)
    CodeType        : Word;           (* Offset = 64 *)
    NoOfTask        : Word;           (* Offset = 66 *)
    Reserved1       : LongInteger;    (* Offset = 68 *)
    Reserved2       : LongInteger;    (* Offset = 72 *)
    Reserved3       : LongInteger;    (* Offset = 76 *)
    ProgramCode     : ARRAY[1..(ActualSize-HeaderSize)] OF BYTE;
                                      (* Offset = 80 *)
End
```

Before a program can be downloaded to the channel, the program controlling download must ensure that the necessary memory area is available, the code-type for the program code is in accordance with the code-type specified for the channel, and that the interpreter program used in the operating system is of the right version.

To download or upload a program, the corresponding command must be sent to the LibraryControl.Command register to initiate the sequence. The program controlling download must wait for LibraryStatus.State = Loading, before the download is executed.

ActualSize for a program is given in LibraryProgramID.ActualSize, following a completed download.

SWNo $D9: ChConfig

The specification of how the selected program should operate after a reset or power failure, is held in ChConfig. This configuration includes a number, specifying which program must be invoked after reset.

ChConfig is a record having the following structure:

```
Record
    Enablebit   : ARRAY[0..7] OF BOOLEAN;     (* Offset = 0 *)
    Functions   : BYTE;                       (* Offset = 1 *)
    Ref_A       : BYTE;                       (* Offset = 2 *)
    Ref_B       : BYTE;                       (* Offset = 3 *)
End
```

Enablebit[0] indicates how the selected program should operate, following a power failure or module reset.

Enablebit[0] = TRUE indicates that the selected program should perform an autostart, resulting in ProgramStatus.State = Running.

Enablebit[0] = FALSE indicates that the selected program should go to ProgramStatus.State = Idle.

The Program Invocation Management on the Program Channel, sends Commands to the selected program after the module reset, to change the State to that specified in Enablebit[0] (Idle or Running). If an error occurs during the autostart procedure, the error code may be read in ChError.His.

Functions is not used.

Ref_A holds the selected program number which is to be invoked after module reset or power-up. No program is selected if Ref_A = 0. Following a successful autostart, Ref_A is copied to ProgramStatus.SelectedProgram.

Ref_B is not used.

SWNo $DA: LibraryControl

LibraryControl is used to set and change the state of a program in the library. The program in the library is chosen using LibraryIndex. Commands can be sent to LibraryControl for controlling a download or upload sequence.

LibraryControl is a record of the following structure:

```
Record
    Command         : BYTE;     (* Offset = 0 *)
    LibraryIndex    : Word;     (* Offset = 2 *)
End
```

Command is used to send a command to the Program Channel to change the state of the program chosen by LibraryIndex. A list of possible commands is given below. The commands and the corresponding numbers conform to the Request Instructions for download used by (MMS).

Command		Purpose
26	InitiateDownloadSequence	Prepare for download. (MMS = InitiateDownloadSequence)
28	TerminateDownloadSequence	Cancel download and end sequence. (MMS = TerminateDownloadSequence)
29	InitiateUploadSequence	Prepare for upload (MMS = InitiateUploadSequence)
31	TerminateUploadSequence	Cancel upload and end sequence. (MMS = TerminateUploadSequence)
36	DeleteProgram	Delete program from the library. (MMS = DeleteProgram)

Command is automatically set to 0 by the Program Channel after writing to the variable. State is immediately updated to one of the corresponding temporary states - Complete or Incomplete, each time a command is sent to the Command variable. When a change in state is made, it is possible to read the LibraryStatus.State variable, to check whether the Command was executed successfully, or that the operation failed.

LibraryIndex chooses one of the programmes in the program library. When a program is chosen, all data relating to this program may be accessed. This data may be read from the variables LibraryProgramID, LibraryStatus and MemoryInfo. Upload and download of the complete program, including the program code, is performed using the IDAndCode variable.

If LibraryIndex is equal to ProgramStatus.SelectedProgram, download is not possible.

The value of LibraryIndex must be in the range from 1 to MaxLibraryIndex. MaxLibraryIndex is found in LibraryStatus. MaxLibraryIndex is 1 in the PD 3260 module.

SWNo $DB: LibraryStatus

LibraryStatus is used to read the current state of a program in the library. The chosen program in the library is defined by LibraryIndex. The max number of programmes in the library is defined by MaxLibraryIndex.

LibraryStatus is a record having the following structure:

```
Record
    State           : BYTE;   (* Offset = 0 *)
    LibraryIndex    : Word;   (* Offset = 2 *)
    MaxLibraryIndex : Word;   (* Offset = 4 *)
End
```

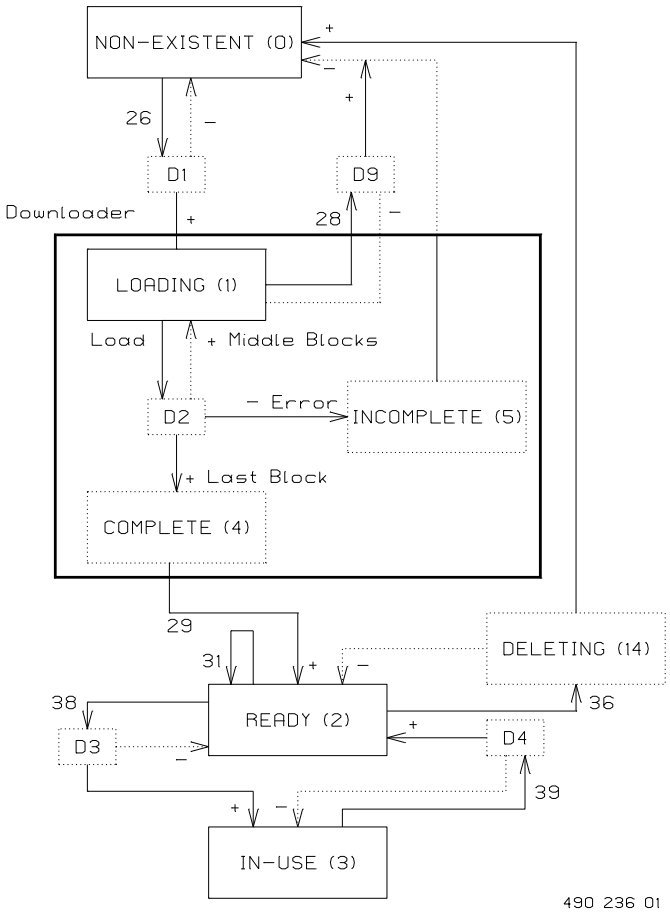
State indicates the current state of the program, eg. loading, ready, non-existent etc. A list of possible states is given below. The states and the corresponding numbers, conform exactly to the states for the download domain used by The Manufacturing Message Specification.

State		**Explanation**
0	Non-existent	No program in this memory type or program segment. (MMS = Non-existent)
1	Loading	Download in progress. (MMS = Loading)
2	Ready	The program is ready to be selected to run. (MMS = Ready)
3	In-use	This program is selected in ProgramControl. The state change to/from In-use is entirely controlled in ProgramControl. Download is not allowed. (MMS = In-use)
4	Complete	The program is completely downloaded and will change state to ready. (MMS = Complete)
5	Incomplete	An error has occurred during download and the program state changes to Non-existent. (MMS = Incomplete)
14	Deleting	Deletion in progress, eg. clearing Flash. (MMS = Deleting)

LibraryIndex is identical to LibraryControl.LibraryIndex.

MaxLibraryIndex indicates the max. number of programmes which can be, or are stored in, the program channel. One type of memory and only one program can be implemented in the PD 3260 module. MaxLibraryIndex therefore always reads 1.

PROGRAM STATE DIAGRAM



+ State transition succeeded - State transition failed.

SWNo $DC: LibraryProgramID

LibraryProgramID corresponds absolutely with ProgramID, but is used to identify the program in the program library. LibraryProgramID is a record of the same type as ProgramID at SWNo $D2.

A program from the library is chosen using a number held in LibraryIndex, which is found in the LibraryControl record.

SWNo $DD: 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.

Maintenance is a Record having the following structure:

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

SWNo $DE: ChType

For the Program Channel, ChType is of the following structure:

```
Record
    ChannelType     : Word;                     (* Offset = 0 *)
    Exist           : Array[0..15] OF Boolean;  (* Offset = 2 *)
    InterpreterVers : Word;                     (* Offset = 4 *)
    CodeType        : Word;                     (* Offset = 6 *)
End
```

For the Program Channel, ChType has the following value:

ChannelType = 11

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	1	0	1	0	0	1	1	1

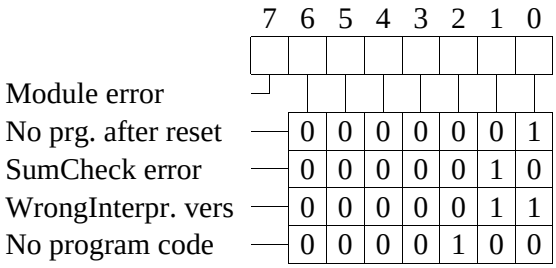
CodeType indicates the kind of program code which can be executed in the Program Channel.

InterpreterVers = 100 (first version)
CodeType = 2 (calculator program code)

SWNo $DF: CHError

```
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 form a combination of an error bit and an error code, having the following meaning. When an error occurs, the corresponding bits are set in both ChError.His and ChError.Act. An UnselectProgram command (39), will clear the error code in ChError.Act.



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

No prg. after reset. This error code can only occur following a reset, caused by one of the following conditions:
If the program number stated in ChConfig.Ref_A=0 and ChConfig.EnableBit[0]=TRUE.
If the program number stated in ChConfig.Ref_A>1 (MaxLibraryIndex).
If the program number stated in ChConfig.Ref_A=1, but the program is non-existent (Actualsize <= 0).

The error code is only set in ChError.His.

SumCheck error. When the program is started, following a Start or Resume command, or an autostart after reset, a sum check is calculated, and compared to the sum check in ProgramID. Any inconsistency will set this error code.

WrongInterpr vers. When the program is started, following a Start or Resume command, or an autostart after reset, the interpreter version in ChType.InterpreterVers is compared with the interpreter version stated in ProgramID. Any inconsistency will set this error code.

No program code. When the program is started, following a Start or Resume command, or an autostart after reset, Actualsize for the program is compared with the headersize. If (Actualsize<=Headersize), this error code will be set.

10 Data Channel (channel E).

A selection of various universal variables are found in the Data Channel. These variables can be used as temporary variables, or for saving calculated results.

Registers on Data Channel . Channel identifier: **DataChannel**

SWNo	Identifier	Memory Type	Read Out	Type
E0	RealA	RAM Auto Save	Decimal	Real
E1	RealB	RAM Init EEPROM	Decimal	Real
E2	IntegerA	RAM Auto Save	Decimal	Integer
E3	WordA	RAM Auto Save	Decimal	Word
E4	BooleanA	RAM Auto Save	Binary	Bit8
E5	BooleanB	RAM Init EEPROM	Binary	Bit8
E6	RealArray	RAM Read Write	Decimal	Array8Real
E7	IntegerArray	RAM Read Write	Decimal	Array8Integer
E8	BooleanArray	RAM Read Write	Binary	Bit32
E9	ChConfig	EEPROM RPW	- - - - - - -	Record
EA	TimerA	RAM Read Write	Decimal	Real
EB	TimerB	RAM Read Write	Decimal	Real
EC	LookUp1	EEPROM RPW	Decimal	Record
ED	LookUp2	EEPROM RPW	Decimal	Record
EE	ChType	PROM Read Only	- - - - - - -	Record
EF	ChError	RAM Read Write	- - - - - - -	Record

SWNo $E0 - $E3:
These are universal variables of simple type. The variables may be used by the calculator program, for storing temporary values or calculated results. Some of these variables are saved in EEPROM at a certain predetermined frequency.

SWNo $E4 - $E8:
These are universal variables of array type, and may be accessed by using the index registers in the calculator.

SWNo $E9: ChConfig
This variable is the same type as the standard channel configuration variable. If an application specific channel is devised, eg. special regulator with ramp up/down facilities, this may be used to select between different configurations.

The ChConfig 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
```

SWNo $EA and $EB: TimerA and TimerB

These registers holds timer variables, which can be used by the calculator program. The timers count down with a resolution of 1/32 second. The count continues through negative values. The timer registers are cleared following a power failure or reset. The maximum value for the timer is approximately 97 days. Following an overflow, the timer continues from the maximum value.

SWNo $EC and $ED: LookUp1 and LookUp2

These variables are declared as lookup tables, having the following format:

```
Coordinate:     Record
                  X:Real;
                  Y:Real;
                End;

    LookUp: Array[1..10] of Coordinate;
```

Each variable represents a line through 10 pairs of X,Y coordinates. LookUp performs a function that returns an interpolated Y value when called with an X value. The X coordinates must be in increasing order. For X values below X1 the function will return the Y1 value and for X values higher than X10, it returns Y10.

SWNo $EE: ChType

For the data channel, ChType has the following structure:

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

The Data Channel is a Company specific channel where ChType has the following value:

ChannelType = $8001

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	1	1	1	1	1	1	1	1

SWNo $EF: ChError

No Error messages will automatically appear in the calculator channel, and therefore ChError normally reads 0. However, the calculator can be programmed to set any of the error bits, to simulate an error condition. This feature may be useful in indicating erroneous data. The register is declared as follows:

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

Any error bit set in ChError will result in the corresponding channel bit to be set in ComHis / ComAct in CommonError in the Service channel.

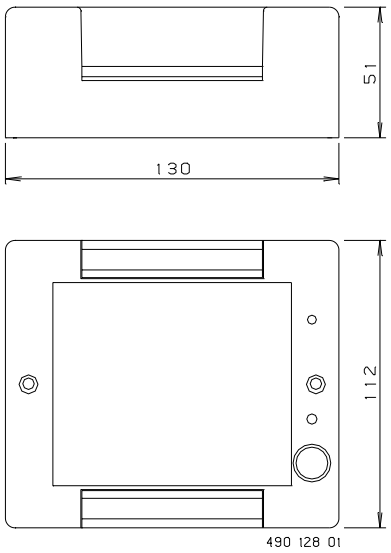
11 Construction, Mechanical.

The PD 3260 module is housed in a black plastic case. The case measures W x H x D = 130.0 x 112.0 x 50.9 mm (tolerance to DIN 16901).

The module is designed for plugging directly onto a mounting rail (EN 50 022 / DIN 46277). The module includes two snap connectors, which provide the terminals for field connection, power and communications.

The module may be DIN rail mounted for a panel mounted configuration, or contained in a sealed box designed for the plant environment. It may be removed for service, without interfering with operational activities on the rest of the network.

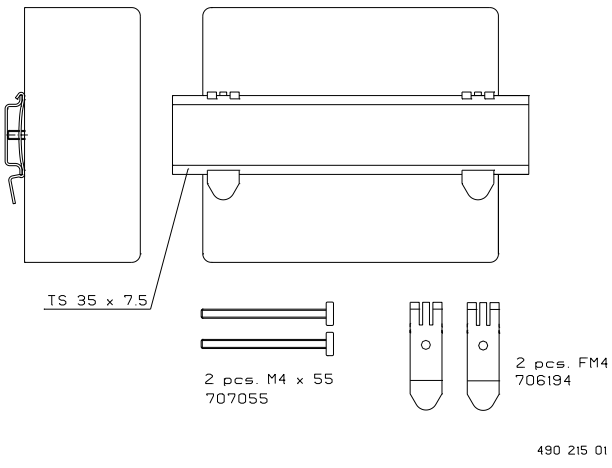
Scale drawing (in mm):



Materials

Case	: Black NORYL GFN (injection moulded)
Front foil	: Polycarbonate.
Back plate	: Black anodized aluminium.
Weight	: 400 gram.

Rail mounting:



12 Specifications.

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

All specifications apply within the approved EMI conditions.

12.1 Power supply.

Power supply DC:	nom 24.0 V
	min 20.0 V
	max 28.0 V
Ripple :	max 5 %
Power consumption (All outputs = OFF):	max 2.0 W
Power consumption (All outputs = ON):	max 3.0 W
Necessary power-up current:	max 400 mA

Fuse 5 A time delay.

12.2 Digital Input.

Input voltage when ON:	< 4.1 V
Input voltage when OFF :	> 15.1 V
Hysteresis:	min 3.2 V
Input current when ON:	max 5.0 mA
Counter frequency for counter in channel:	max 50 Hz

12.3 Digital Output.

Load current when ON (Sink):	max 1.0 A
Leakage current when OFF:	max 0.5 mA
Short circuit cut-off delay time (cut-off at output current > 2 A):	max 10 ms
One-shot and duty-cycle resolution :	125 ms
FeedBack update time:	125 ms

Load current measurement :

Accuracy :	± 19 mA
Resolution :	12.5 mA
Repeatability :	± 12.5 mA
Current measurement update time :	125 ms

12.4 Power Monitor.

Variable update time at 50 Hz (five periods of line frequency):	100 ms
Input current measurement:	
Autorange:	nom 1.0 / 0.2 A RMS
Accuracy:	max ±0.5 %
Max. input:	±2A peak
Input resistance	nom 0.2 Ω
Input voltage measurement:	
Range:	nom 24 V RMS
Accuracy:	max ±0.5 %
Max. input:	±50V peak
Input resistance	nom 50 kΩ
Power factor measurement:	
Range:	-1.00 to +1.00
Accuracy:	max. ± 1 %
Line frequency measurement:	
Range:	45 - 62 Hz
Accuracy:	± 0.1 Hz

12.5 Generator Switch.

Tacho input:	
Range:	0-100 Hz
Accuracy:	± 0.1 Hz

12.6 Thyristor Switch.

Modulation frequency:	10 kHz
Coupling degree:	0 - 180 deg
Update frequency (at 50 Hz line frequency):	20 ms

12.7 Calculator program.

Memory size:	7000 bytes
Instruction time:	typical 0.4 ms

12.8 Ambient Temperature.

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

12.9 Humidity.

Relative humidity :	max 95 %

12.10 Approvals.

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

13 Survey of variables in the PD 3260 module.

SWNo	Service 0	Digital_IO_x 1-8	CommonIO 9	PowerMonitor A	GeneratorSwitch B	ThyristorSwitch C	Calculator D	DataChannel E
x0	NumberOfSWNo	FlagReg	OutFlags	Power	FlagReg	FlagReg	ProgramControl	RealA
x1	DeviceID	OutTimer	InFlags	Energy	Phase	TrigAngle	ProgramStatus	RealB
x2		Counter	IOChError	PhasePower	PhaseSpeed		ProgramID	IntegerA
x3	Reset	OutCurrent		PhaseEnergy	PhaseAcc			WordA
x4	PnetSerialNo	OperatingTime		Current				BooleanA
x5		UserByteArray		Voltage			SystemPointer	BooleanB
x6		FBTimer		PowerFactor		TrigSlope		RealArray
x7	FreeRunTimer	FBPreset		HighLevel			MemoryInfo	IntegerArray
x8	WDTimer	OutPreset		LowLevel			IDAndCode	BooleanArray
x9	ModuleConfig	ChConfig		ChConfig	ChConfig	ChConfig	ChConfig	ChConfig
xA	WDPreset	MinCurrent		LineFreq	TachoScale		LibraryControl	TimerA
xB		MaxCurrent		FullScale	StartSpeed	TrigMax	LibraryStatus	TimerB
xC		UserRealArray		Factors	DelayComp	TrigMin	LibraryProgramID	LookUp1
xD	WriteEnable	Maintenance		Maintenance	Maintenance	Maintenance	Maintenance	LookUp2
xE	ChType	ChType	ChType	ChType	ChType	ChType	ChType	ChType
xF	CommonError	ChError	ChError	ChError	ChError	ChError	ChError	ChError

Contents