

# LD-IV and LD-TM Large Digit Display Output Addendum

---

*AMALGAMATED INSTRUMENT CO*

*ABN: 80 619 963 692*

*Unit 5, 28 Leighton Place Hornsby    Telephone: +61 2 9476 2244    e-mail: sales@aicpl.com.au*  
*NSW 2077 Australia                      Facsimile: +61 2 9476 2902    Internet: www.aicpl.com.au*

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Setting up the relay PI controller</b>	<b>4</b>
<b>3</b>	<b>Analog PI control output</b>	<b>14</b>
<b>4</b>	<b>Serial communication</b>	<b>22</b>
<b>5</b>	<b>Serial operation</b>	<b>24</b>
<b>6</b>	<b>Modbus operation</b>	<b>28</b>
<b>7</b>	<b>Data logger</b>	<b>31</b>

# 1 Introduction

This addendum to the main LD-IV and LD-TM manual contains information for the operation of the PI control mode for relay and analog output operation and for serial communications. Refer to the main manual for all other information regarding this instrument.

The standard and optional outputs available are as follows:-

- Four output relays are fitted as standard. This addendum covers the use of relays 1 and 2 as PI control relays. Refer to the standard manual for details of the standard on/off alarm or control operation.
- Optional isolated dual 4-20mA output. This addendum covers the use of output 1 as a PI control output. Refer to the standard manual for details of outputs 1 and 2 used as retransmission outputs.
- Optional serial communications. This addendum covers the use of the optional RS232 or RS485 serial communications in both standard ASCII and Modbus RTU modes.
- Datalogger. This addendum covers the datalogger commands for use with polling software other than the supplied download software. For information on using the datalogger with the download software issued with the datalogger consult the separate “Download Software User Guide”.

## 2 Setting up the relay PI controller

The Relay Proportional + Integral Controller can be made to operate in either pulse width control or frequency control mode via the **Rx OPER** function. Note that the **Rx OPER** function will not be seen until a value has been set for the low or high alarm e.g. for **R ILo** or **R IH**. The best results are usually achieved by initially configuring as a “Proportional Only” controller and then introducing the Integral functions when stable results are obtained.

Relay 1 and relay 2 can be set to operate in PI control mode. Any other relays fitted will only operate in normal, non PI operation. The “x” in the **Rx OPER** and other functions indicates the chosen relay i.e. for relay 1 the display will show **R 1 OPER**, **R 1 SP** etc. The **Rx OPER** function allows three choices of operating mode for the chosen relay, namely **Rx.AL**, **Rx.tP** and **Rx.Fr**. If **Rx.AL** is selected the chosen relay will operate as a setpoint relay whose operation is controlled by the **RxH**, **RxLo** etc. settings and the PI control settings will not be seen. See the “Explanation of functions” chapter for details of operation when **Rx.AL** is selected. If **Rx.tP** is selected then the chosen relay will operate in pulse width control mode. If **Rx.Fr** is selected then the chosen relay will operate in the frequency control mode.

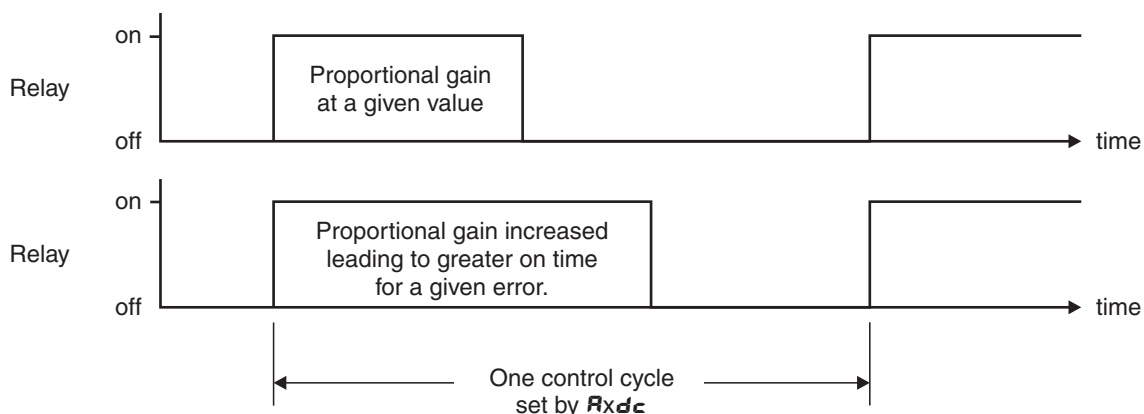
**Pulse width control** - operates by controlling the on to off time ratio of the relay. In a typical application this would be used to control the length of time for which a dosing pump is switched on during a control cycle i.e. the pump or other device will continuously operate for the length of time the relay is activated and will stop operating when the relay is de-activated.

**Frequency control** - operates by changing the rate at which the relay switches on and off. In a typical control application the frequency control operation is particularly suited for use when one shot dosing is used i.e. the pump or other device puts out a fixed dosing quantity for every pulse received.

### 2.1 Relay pulse width modulation control mode

To use pulse width modulation control **Rx.tP** must be selected at the **Rx OPER** function.

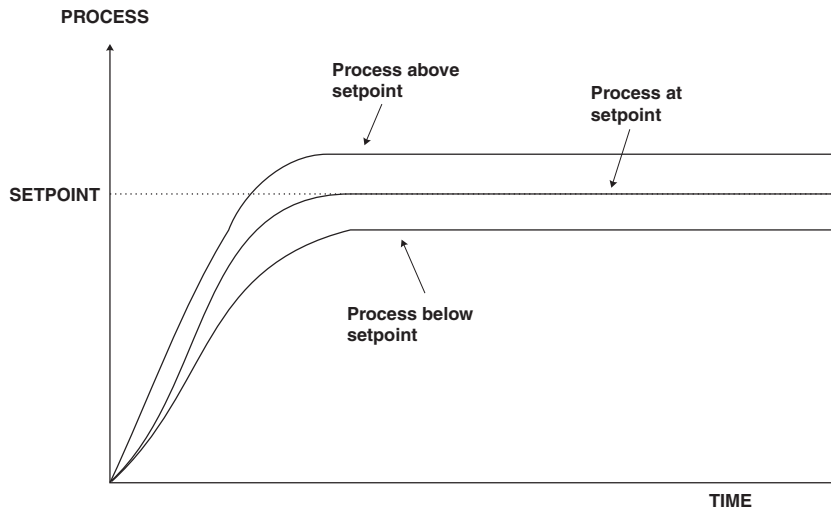
#### Pulse width control



## 2.2 PI relay control setpoint

**Display:** *Rx.SP*  
**Range:** Any display value  
**Default Value:** **0**

The control setpoint is set to the value in displayed engineering units required for the control process. The controller will attempt to vary the control output to keep the process variable at the setpoint. Note that the control setpoint value can be reached and adjusted via the “easy access” mode (see “Explanation of functions” chapter) if the **ACCS** function is set to **EASY**. This feature could be useful if the setpoint is to be frequently changed.



## 2.3 PI relay control span

**Display:** *ctrl SPAN*  
**Range:** Any display value  
**Default Value:** **100**

The function of the control span is to define the limit to which the PI control values will relate. The control span value will be common to all control relays i.e. if more than one control relay output is being used then each of these relays operates from the same control span setting. The span value defines the range over which the input must change to cause a 100% change in the control output when the proportional gain is set to **1.000**. This function affects the overall gain of the controller and is normally set to the process value limits that the controller requires for normal operation. For example if the control setpoint (*Rx.SP*) is **70** and the *ctrl SPAN* is **20** and *Rx.P9* is set to **1.000** then an error of **20** from the setpoint will cause a 100% change in proportional control output. For example with *Rx.SP* at **70**, *ctrl SPAN* at **20**, *Rx.P9* at **1.000** and *Rx.b5* at **0.000** a display reading of **50** or lower (*Rx.SP* minus *ctrl SPAN*) the control output will be at 100% i.e. the relay will be on continuously. The control output will then gradually adjust the on/off time as the display value reaches the setpoint.

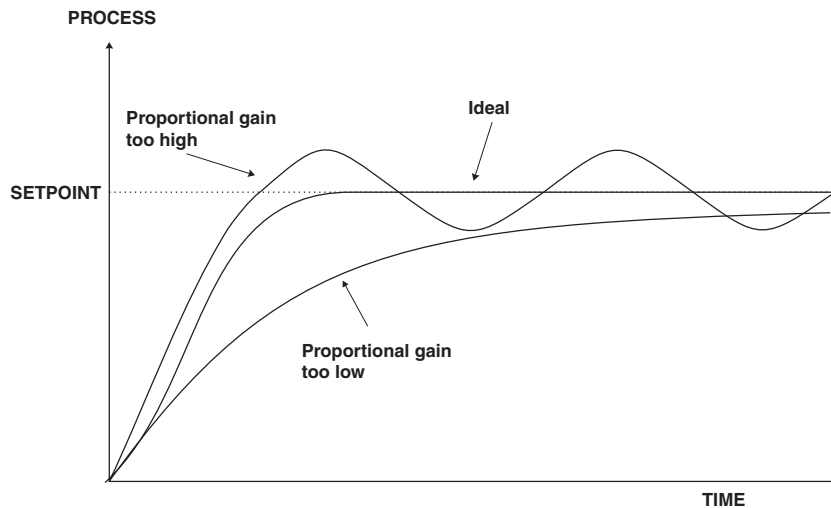
For instruments with more than one input where the number of decimal points displayed may vary the control span will take on the value of the main display and so may or may not match the decimal points shown in the input being controlled. e.g. a control span of 2.00 will act as a control span of 20.0 if the input to be controlled is displayed with only 1 decimal point.

## 2.4 PI relay proportional gain

Display: **Rx.P9**  
Range: **-32.767 to 32.767**  
Default Value: **0.0 10**

Note: the range value may be restricted if the number of display digits does not allow viewing of the full range.

The proportional value will determine the degree to which the controller will respond when there is a difference (error) between the measured value and the process setpoint. If the proportional gain is increased then for a given error the relay on time will be increased (or decreased if the error is on the other side of the setpoint). The proportional gain action can be reversed by setting a negative gain i.e. with a negative gain the on time will reduce as the error increases. With a proportional gain of **1.000** and an error of **10** or more (with control span set at **10**) the controller will increase the frequency by 100% if possible. With a proportional gain of **0.500** an error of **10** or more (with control span set at **10**) will cause the controller to increase the frequency by 50%, if possible. Too much proportional gain will result in instability due to excessive overshoot of the setpoint. Too little proportional gain will lead to a slow response.



This table shows the effect of the output frequency of changing proportional gain and bias with the following settings:

$$\text{ctrl SPAN} = 20, \text{A 1.dz} = 1.0, \text{A 1.1 9} = 0.000$$

<b>A 1.SP</b>	<b>A 1.P9</b>	<b>A 1.b5</b>	<b>Effect on relay operation</b>
<b>70</b>	<b>1.000</b>	<b>0.0</b>	Reading of <b>50</b> or below - relay permanently on. Reading of <b>50</b> to <b>70</b> - relay pulses with off time increasing as value approaches <b>70</b> . Reading <b>70</b> or above - relay permanently off.
<b>70</b>	<b>1.000</b>	<b>100.0</b>	Reading of <b>70</b> or below - relay permanently on. Reading of <b>70</b> to <b>90</b> - relay pulses with off time increasing as value approaches <b>90</b> . Reading <b>90</b> or above - relay permanently off.
<b>70</b>	<b>1.000</b>	<b>50.0</b>	Reading of <b>60</b> or below - relay permanently on. Reading of <b>60</b> to <b>70</b> - relay pulses with off time increasing as value approaches <b>70</b> . Reading <b>70</b> - relay pulses at 50% on and 50% off. Reading <b>70</b> to <b>80</b> - relay pulses with off time increasing as value approaches <b>80</b> . Reading <b>80</b> or above - relay permanently off.
<b>70</b>	<b>0.500</b>	<b>50.0</b>	Reading <b>50</b> or below - relay permanently on. Reading <b>50</b> to <b>70</b> - relay pulses with off time increasing as value approaches <b>70</b> . Reading <b>70</b> - relay pulses at 50% on and 50% off. Reading <b>70</b> to <b>90</b> - relay pulses with off time increasing as value approaches <b>90</b> . Reading <b>90</b> or above - relay permanently off.
<b>70</b>	<b>- 1.000</b>	<b>50.0</b>	Reading of <b>60</b> or below - relay permanently off. Reading of <b>60</b> to <b>70</b> - relay pulses with on time increasing as value approaches <b>70</b> . Reading <b>70</b> - relay pulses 50% on and 50% off. Reading <b>70</b> to <b>80</b> - relay pulses with on time increasing as value approaches <b>80</b> . Reading <b>80</b> or above - relay permanently on.

## 2.5 PI relay integral gain

Display: **Ax: 9**

Range: **-32.767 to 32.767**

Default Value: **0.000**

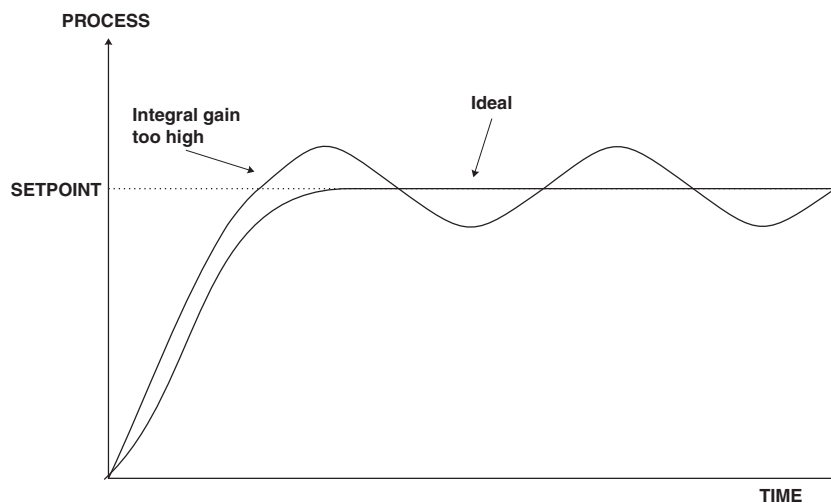
Note: the range value may be restricted if the number of display digits does not allow viewing of the full range.

The Integral action will attempt to correct for any offset which the proportional control action is

unable to correct (e.g. errors caused by changes in the process load). When the integral gain is correctly adjusted the control output is varied to maintain control by keeping the process variable at the same value as the control setpoint. Since the integral gain is time based the output will gradually increase if the error does not decrease i.e. if the measured value remains constant and there is an error (a difference between the measured value and the setpoint) then the frequency will be increased compared to the previous frequency output. The higher the proportional gain, the greater the degree by which the on to off ratio will be affected i.e. the response will be greater at higher integral gain settings. With an integral gain of **1.000** an error of **10** or more (with control span set at **10**) will cause the integral action to try to correct at the rate of 100% minute. With an integral gain of **0.200** an error of **10** or more will cause the integral action to try to correct at the rate of 20% per minute. Too high an integral gain will result in instability. Too low an integral gain will slow down the time taken to reach the setpoint. The optimum setting will depend on the lag time of the process and the other control settings. Start with a low figure (e.g. **0.200**) and increase until a satisfactory response time is reached. The integral gain figure has units of gain/minute. The integral action can be reversed by setting a negative gain figure, note that the sign of the integral gain must match the sign of the proportional gain. The integral control output follows the formula:

$$\text{Integral control output} = \frac{\text{error} \times I_g \times \text{time (seconds)}}{60} + \text{previous integral control output}$$

Where  $I_g$  is the integral gain set via **Rx: I 9**.



## 2.6 PI relay integral control high limit

Display: **Rx: I H**  
 Range: **0.0 to 100.0**  
 Default Value: **100.0**

The maximum limit can be used to reduce overshoot of the control setpoint when the control output is increasing i.e. rising above the setpoint. Other than this the limit operates in the same manner as the low limit described previously.



## 2.7 PI relay integral control low limit

**Display:** ***Ax.l***  
**Range:** **0.0 to 100.0**  
**Default Value:** **100.0**

The minimum limit can be used to reduce overshoot of the control setpoint when the control output is being reduced i.e. falling below the setpoint. The low limit reduces the available output swing by a percentage of the maximum output. Without a limit the integral output can be very large at the time the setpoint is reached and a large overshoot of the will then result. Settings available are from **0.0** to **100.0** (%). If the limit setting is too high then overshoot will result. If the setting is too low then the integral output can be limited to such an extent that the setpoint cannot be maintained.

Start with a low value such as **20.0** and increase or decrease the value until a satisfactory result is obtained. The advantage of using separate low and high limits is that in many applications the response is very one directional e.g. the system may respond very quickly to a heat input but may cool down at a much slower rate. Separate high and low limit settings allow independent limiting of the integral control swing below and above the setpoint so a smaller minimum limit can be set to limit swings below the setpoint to compensate for the slower cooling time.

The minimum and maximum limits are used in conjunction with the output bias setting to maintain the control process setpoint value. For example with a bias (***Ax.bs***) set at 50%, minimum limit set at 20% and a maximum limit of 30% the actual bias when the process is at the setpoint may be anywhere between 30% and 80% i.e. Integral control is being used to alter the bias setting in order to maintain the process at the setpoint. In this case the minimum term will allow the bias to drop to a value between 50% and 30% in order to maintain the setpoint. The maximum term will allow the bias point to rise to a value between 50% and 80% in order to maintain the setpoint.

## 2.8 PI relay control output bias

**Display:** ***Ax.bs***  
**Range:** **0.0 to 100.0**  
**Default Value:** **50.0**

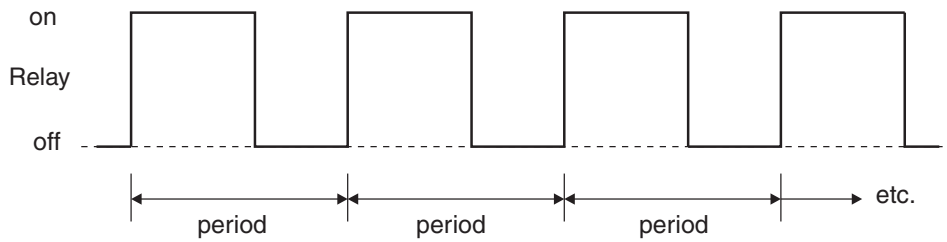
The control bias sets the ideal steady state output required once the setpoint is reached. Settings are in % from **0.0** to **100.0**. When set at **0.0** the relay will be de-activated for the entire control period when the measured input is at the setpoint (depending on proportional and integral gain settings). If set at **50.0** then the relay operation frequency will on for 50% and off for 50% of the duty cycle time when the measured input is at the setpoint. If set at **100.0** then the relay will activated for the whole time whist the measured input is at the setpoint.

## 2.9 PI relay control cycle period

**Display:** ***Ax.dc***  
**Range:** **0 to 250**  
**Default Value:** **10**

Displays and sets the control period cycle from **0** to **250** seconds. The control period sets the total time for each on/off cycle. This time should be set as long as possible to reduce wear of the

control relay and the controlling device.



## 2.10 Setting up the PI pulse width controller

1. Set the ***Ax OPEr*** function to ***Ax.tP***.
2. Set the control setpoint ***Ax.SP*** to the required setting.
3. Set the control span ***ctrl SPAN*** to the required setting.
4. Set the proportional gain ***Ax.P9*** to an arbitrary value e.g. ***0.500***.
5. Set the integral gain ***Ax.I 9*** to ***0.000*** (i.e. off).
6. Set the low and high integral ***Ax.I L*** and ***Ax.I H*** limits to an arbitrary value e.g. ***20.00***.
7. Set the bias ***Ax.b5*** to ***50.0***.
8. Set the cycle ***Ax.dc*** period to ***20*** seconds.

Initialise the control system and monitor the control results. If the original settings causes process oscillations then gradually decrease the proportional gain until the oscillations decrease to an acceptable steady cycle. If the original settings do not cause process oscillations then gradually increase the proportional gain until a steady process cycling is observed.

Once the steady cycling state is achieved note the difference between the display value and the control setpoint value. Gradually increase or decrease the bias value until the displayed value matches (or cycles about) the control setpoint value.

Gradually increase the integral gain until the process begins to oscillate. Then reduce the integral gain slightly to regain the control without this added oscillation.

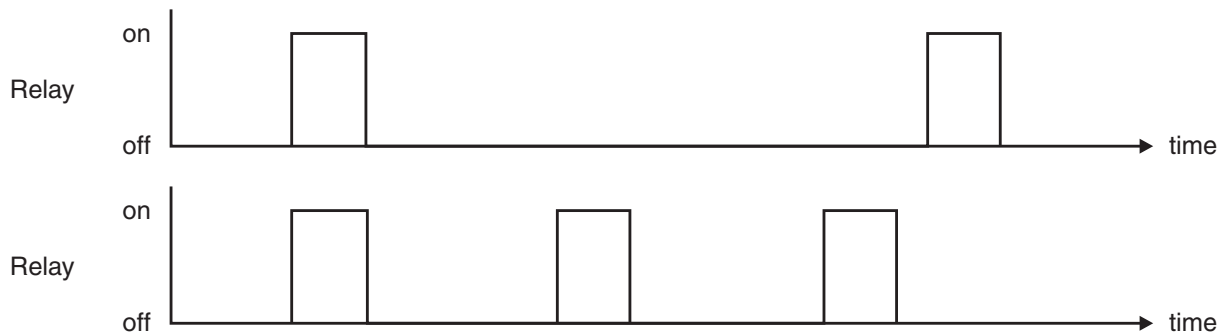
Create a step change to the process conditions and observe the control results. It may be necessary to fine tune the settings and use integral limits to obtain optimum results.

Set up sequence	Symptom	Solution
Proportional gain	Slow response	Increase proportional gain
Proportional gain	High overshoot or oscillation	Decrease proportional gain
Proportional bias	Process above or below setpoint	Increase or decrease bias as required
Integral gain	Slow response	Increase integral gain
Integral gain	Instability or oscillations	Decrease integral gain

## 2.11 Relay frequency modulation control mode

To use pulse width modulation control **Rx.Fr** must be selected at the **Rx OPER** function. In frequency modulation mode the relay on time is fixed. A minimum relay off time can also be set. The control program will vary the actual off time to suit the error seen between the setpoint and the measured temperature at the time. For example if extra dosing is needed to reach the setpoint then the off time will be reduced resulting in more on pulses per period of time i.e. the frequency of the pulses is controlled to allow the setpoint to be maintained.

Frequency control - pulse frequency varies according to settings and control requirement



Frequency PI control operation has many functions in common with PI pulse width control, refer to the appropriate sections as shown below for these common functions.

**Rx.SP** (Control setpoint) - refer to section 2.2

**ctrl SPAN** (Control span) - refer to section 2.3

**Rx.PG** (Proportional gain) - refer to section 2.4

**Rx.I G** (Integral gain) - refer to section 2.5

**Rx.I L** (Integral control low limit) - refer to section 2.7

**Rx.I H** (Integral control high limit) - refer to section 2.6

**Rx.bs** (PI control bias) - refer to section 2.8

**Rx.dc** (PI control cycle period) - refer to section 2.9. In frequency mode this function sets the minimum off time. If set to **0** the relay will be disabled. The control program can extend the off time to maintain the setpoint but not reduce it. If a 100% error is seen then the pulse rate will be at its maximum i.e. the off time will equal **Rx.dc**. If a 50% error is seen there will be a pulse every 2 times **Rx.dc**. For a 25% error there will be a pulse every 4 times **Rx.dc** and for a 10% error there will be a pulse every 10 times **Rx.dc**.

This table shows the effect of the output frequency of changing proportional gain and bias with the following settings:

$$ctrl\ SPAN = 20, R\ i_{dc} = 1.0, R\ i_{19} = 0.000$$

<b><i>R i.SP</i></b>	<b><i>R i.P9</i></b>	<b><i>R i.b5</i></b>	<b>Effect on relay operation</b>
<b>70</b>	<b>1.000</b>	<b>0.0</b>	Reading of <b>50</b> or below - relay pulses at maximum frequency. Reading of <b>50</b> to <b>70</b> - relay pulses with frequency decreasing as value approaches <b>70</b> . Reading <b>70</b> or above - relay permanently off.
<b>70</b>	<b>1.000</b>	<b>100.0</b>	Reading of <b>70</b> or below - relay pulses at maximum frequency. Reading of <b>70</b> to <b>90</b> - relay pulses with frequency decreasing as value approaches <b>90</b> . Reading <b>90</b> or above - relay permanently off.
<b>70</b>	<b>1.000</b>	<b>50.0</b>	Reading of <b>60</b> or below - relay pulses at maximum frequency. Reading of <b>60</b> to <b>80</b> - relay pulses with frequency decreasing as value approaches <b>80</b> . (period increased by 50% at <b>70</b> compared to minimum period e.g. if minimum period is 4 seconds the period at <b>70</b> will be 6 seconds) Reading <b>80</b> or above - relay permanently off.
<b>70</b>	<b>0.500</b>	<b>50.0</b>	Reading <b>50</b> or below - relay pulses at maximum frequency. Reading <b>50</b> to <b>90</b> - relay pulses with frequency decreasing as value approaches <b>90</b> . (period increased by 50% at <b>70</b> compared to minimum period e.g. if minimum period is 4 seconds the period at <b>70</b> will be 6 seconds) Reading <b>90</b> or above - relay permanently off.
<b>70</b>	<b>- 1.000</b>	<b>50.0</b>	Reading of <b>60</b> or below - relay permanently off. Reading of <b>60</b> to <b>80</b> - relay pulses with frequency decreasing as value approaches <b>80</b> . (period increased by 50% at <b>70</b> compared to minimum period e.g. if minimum period is 4 seconds the period at <b>70</b> will be 6 seconds) Reading <b>80</b> or above - relay pulses at maximum frequency.

## 2.12 PI relay on duration

Display: ***Rx.dr***

Range: **0.0 to 25.0**

Default Value: **1.0**

Displays and sets the control relay on duration from **0.0** to **25.0** seconds. If set to **0.0** the relay will be disabled. The duration should be long enough to ensure that the device being controlled receives an acceptable on pulse.

## 2.13 Setting up the PI frequency controller

1. Set the ***Rx OPEr*** function to ***RxFr***.
2. Set the control setpoint ***Rx.SP*** to the required setting.

3. Set the control span **ctrl: SPAN** to the required setting.
4. Set the proportional gain **Ax.P9** to an arbitrary value e.g. **0.500**.
5. Set the integral gain **Ax.I 9** to **0.000** (i.e. off).
6. Set the low and high integral **Ax.I L** and **Ax.I H** limits to an arbitrary value e.g. **20.00**.
7. Set the bias **Ax.b5** to **50.0**.
8. Set the cycle **Ax.dc** period to **20** seconds.
9. Set the relay on time **Ax.dr** to an arbitrary value e.g. **1.0**

Initialise the control system and monitor the control results. If the original settings causes process oscillations then gradually decrease the proportional gain until the oscillations decrease to an acceptable steady cycle. If the original settings do not cause process oscillations then gradually increase the proportional gain until a steady process cycling is observed.

Once the steady cycling state is achieved note the difference between the display value and the control setpoint value. Gradually increase or decrease the bias value until the displayed value matches (or cycles about) the control setpoint value.

Gradually increase the integral gain until the process begins to oscillate. Then reduce the integral gain slightly to regain the control without this added oscillation.



Create a step change to the process conditions and observe the control results. It may be necessary to fine tune the settings and use integral limits to obtain optimum results.

Set up sequence	Symptom	Solution
Proportional gain	Slow response	Increase proportional gain
Proportional gain	High overshoot or oscillation	Decrease proportional gain
Proportional bias	Process above or below setpoint	Increase or decrease bias as required
Integral gain	Slow response	Increase integral gain
Integral gain	Instability or oscillations	Decrease integral gain

## 3 Analog PI control output

PI control functions will only be seen if PI control software is available for the instrument and if the optional isolated analog or dual isolated analog output is fitted. In dual analog instruments only the first output can be set for PI control output.

The PI (proportional + integral) control output may be configured for proportional only (i.e. integral gain set to 0.000) or proportional + integral control. The control output may be link selected as either a 4-20mA, 0-1VDC or 0-10VDC signal. Using the control function settings described below the instrument will vary the control output signal in such a way that the process being monitored is kept as close as possible to the control setpoint. The control may be turned on or off via the **FEECtrl** function. When the **FEECtrl** function is set to **OFF** the output will act as a retransmission output rather than a control output and the PI control functions will not be seen. When set to **on** the PI control functions will be seen but the standard retransmission functions (e.g. **FEE-** and **FEE+**) will not.

The  or  buttons may be used to view the control setpoint when PI control is used. The best PI control results are usually achieved by initially configuring as a proportional only controller and introducing the Integral control once stable results have been obtained from proportional only control.

### 3.1 Proportional control output

For proportional only control the output is found from:

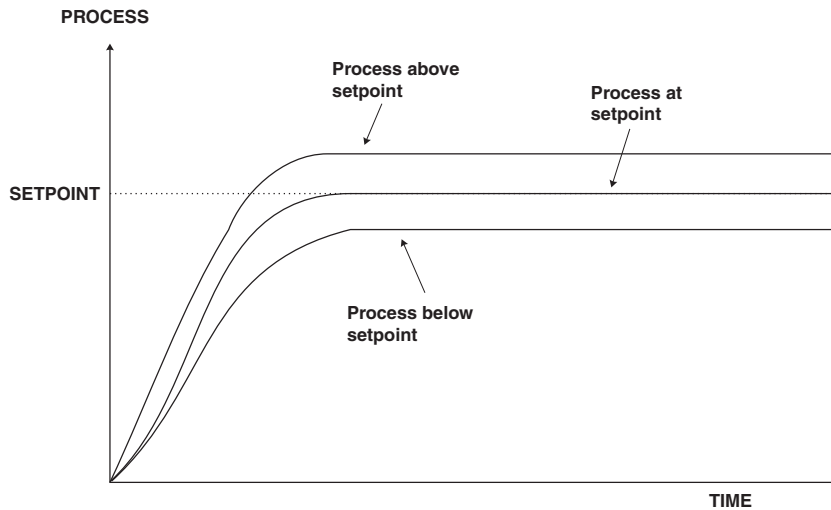
$$\text{Proportional control output} = \text{Error} \times \text{Proportional gain} + \text{Offset}$$

Where the Error is defined by the **C.SPn** function, the Proportional gain is set by the **C.PG** function and the Offset is set by the **C.PO** function.

### 3.2 PI analog control setpoint

**Display:** **C.SET**  
**Range:** Any display value  
**Default Value:** **0**

The control setpoint is set to the value in displayed engineering units required for control of the process. The controller will attempt to vary the control output to keep the process variable at the setpoint. Note that the control setpoint can be made available in **FUNC** mode and in some cases in “easy access” mode via the **FEE SPAC** function described in this chapter.



### 3.3 PI analog control on or off

**Display:** `FEC ctrl`  
**Range:** `on` or `OFF`  
**Default Value:** `OFF`

This function determines whether the analog output will be used as a PI control output or as a retransmission output. When `on` is selected the analog output will be used as a control output, all of the control functions will be seen but no analog retransmission functions will be seen. When set to `OFF` the analog output will be used as a retransmission output, the retransmission functions, such as `FEC`, `FEC-` and `FEC+` will be seen and the control functions will not appear on the display.

### 3.4 PI analog control span

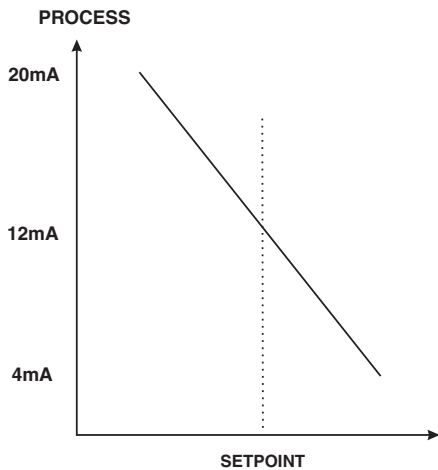
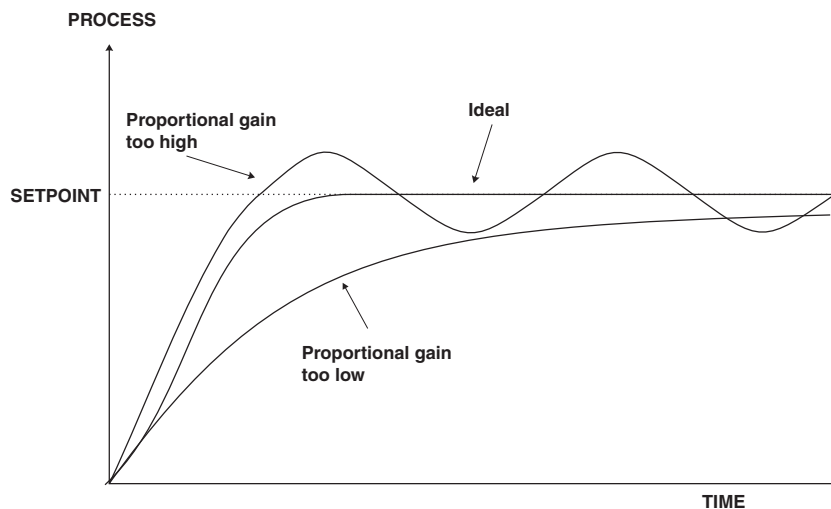
**Display:** `C.SPAN`  
**Range:** `0` to any positive display value  
**Default Value:** `0`

The function of the control span is to define the limit to which the proportional control values will relate. The span value defines the range over which the input must change to cause a 100% change in the control output when the proportional gain is set to 1.000. This function affects the overall gain of the controller and is normally set to the process value limits that the controller requires for normal operation. For example if the control setpoint (`C.SET`) is `50.0` and the `C.SPAN` is `15` then an error of 15 from the setpoint will cause a 100% change in proportional control output. For example, assuming that the control output is a 4-20mA signal, with `C.SET` at `50.0`, `C.SPAN` at `15.0`, `C.PG` at `1.000` and `C.PO` at `0.000` a display reading of `35.0` or lower (`C.SET - C.SPAN`) the control output will be at 100% i.e. 20mA. The control output will then gradually fall as the display value reaches the setpoint.

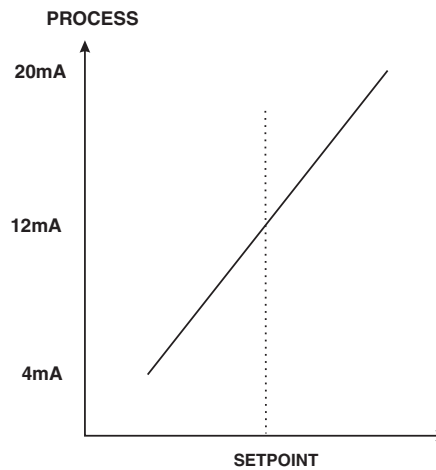
### 3.5 PI analog proportional gain

Display: **C.P9**  
Range: **-32.767** to **32.767** number of display digits allowing  
Default Value: **0.000**

The proportional gain is the ratio between the change in measured input and change in control output. Too much proportional gain will result in instability. Example 1, if the proportional gain is set to **1.000** and the measured input changes by 100% of the span set in **C.SPn** then the output will change by 100%. Example 2, if the proportional gain is set to **2.000** and the measured input changes by 50% of the range set in **C.SPn** then the output will change by 100%. Example 3, if the proportional gain is set to **2.000** and the measured input changes by 25% of the range set in **C.SPn** then the output will change by 50%. Setting a negative proportional gain will reverse the control output.



Positive **C.P9** value e.g. **1.000**



Negative **C.P9** value e.g. **-1.000**



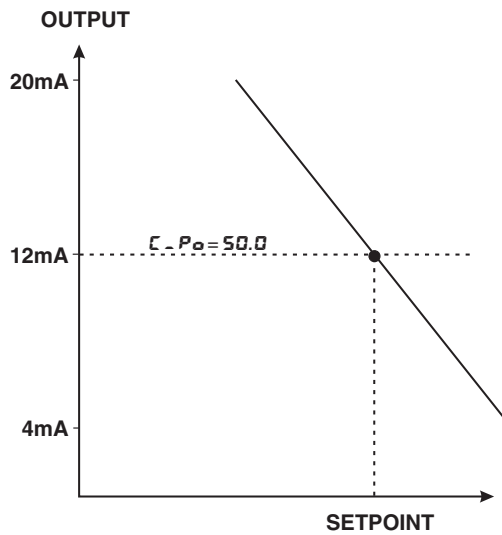
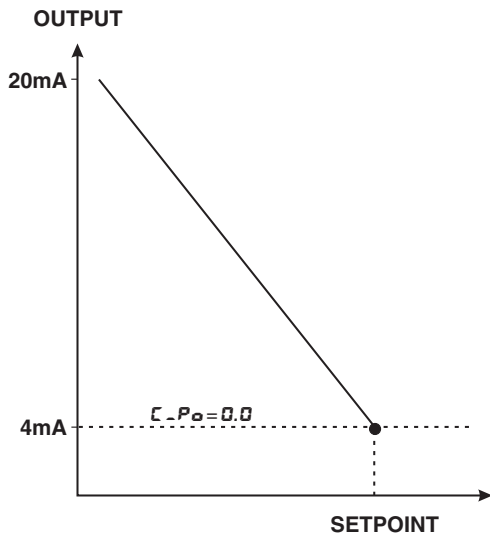
This table shows the effect of the output current of changing proportional gain and offset with the following settings: **C.SPn = 2.00, C.I 9 = 0.000**

<b>C.SET</b>	<b>C.P9</b>	<b>C.Po</b>	<b>Effect on analog output (4-20mA used in this example)</b>
<b>7.00</b>	<b>1.000</b>	<b>0.0</b>	Reading of 5.00 or below - 20mA output Reading of 5.00 to 7.00 - mA output decreasing as reading approaches 9.00 Reading 7.00 or above - 4mA output
<b>7.00</b>	<b>1.000</b>	<b>100.0</b>	Reading of 7.00 or below - 20mA output Reading of 7.00 to 9.00 - mA output decreasing as reading approaches 9.00 Reading 9.00 or above - 4mA output
<b>7.00</b>	<b>1.000</b>	<b>50.0</b>	Reading of 6.00 or below - 20mA output Reading of 6.00 to 8.00 - mA output decreasing as reading approaches 8.00 with 12mA output at 7.00 Reading 8.00 or above - 4mA output
<b>7.00</b>	<b>0.500</b>	<b>50.0</b>	Reading 5.00 or below - 20mA output Reading 5.00 to 9.00 - mA output decreasing as reading approaches 9.00 with 12mA output at 7.00 Reading 9.00 or above - 4mA output
<b>7.00</b>	<b>- 1.000</b>	<b>50.0</b>	Reading of 6.00 or below - 4mA output Reading of 6.00 to 8.00 - mA output increasing as reading approaches 8.00 with 12mA output at 7.00 Reading 8.00 or above - 20mA output

### 3.6 PI analog proportional offset %

Display: **C.Po**  
 Range: **0.0** to **100.0**  
 Default Value: **0.0**

The proportional offset is initially used to set the output value when operating the instrument as a proportional only controller. The proportional offset determines what % of the proportional control output will be given when the process value reaches the setpoint value. If set to **0.0** then there will be zero output (e.g. 4mA for a 4-20mA output) when the process value reaches the setpoint value. If set to **50.0** then there will be a 50% output (e.g. 12mA for a 4-20mA output) when the process reaches the setpoint value. If set to **100.0** then there will be a 100% output (e.g. 20mA for a 4-20mA output) when the process reaches the setpoint value. If using proportional only control then when stable control is established there may be a difference between the process and the setpoint values. By altering the proportional offset value the difference may be minimised.



### Proportional only control examples

For a 4-20mA control output (0% = 4mA and 100% = 20mA) the setpoint is 7.0, the span is 2.0, the proportional gain is 1.000 and the offset is 0.0. If the reading on the display is 6.8 then the error is 10% (i.e. 10% of the span figure).

$$\text{Proportional control output} = \text{Error} \times \text{Proportional gain} + \text{Offset}$$

$$\text{Proportional control output} = 10\% \times 1 + 0\% = 10\% \text{ or } 5.6\text{mA}$$

If the proportional gain were to be changed to 2.000 then:

$$\text{Proportional control output} = 10\% \times 2 + 0\% = 20\% \text{ or } 7.2\text{mA}$$

If the proportional gain were to be changed to 0.500 then:

$$\text{Proportional control output} = 10\% \times 0.5 + 0\% = 5\% \text{ or } 4.8\text{mA}$$

If the offset were now to be changed to 50.0 (50%) then:

$$\text{Proportional control output} = 10\% \times 2 + 50\% = 55\% \text{ or } 12.8\text{mA}$$

## 3.7 Integral control output

The integral control output can be found from:

$$\text{Integral control output} = \frac{\text{Error} \times \text{IG} \times \text{time}(\text{secs})}{60} + \text{previous integral control output}$$

Where IG is the integral gain is set by the **E: 9** function.

## 3.8 PI analog integral gain

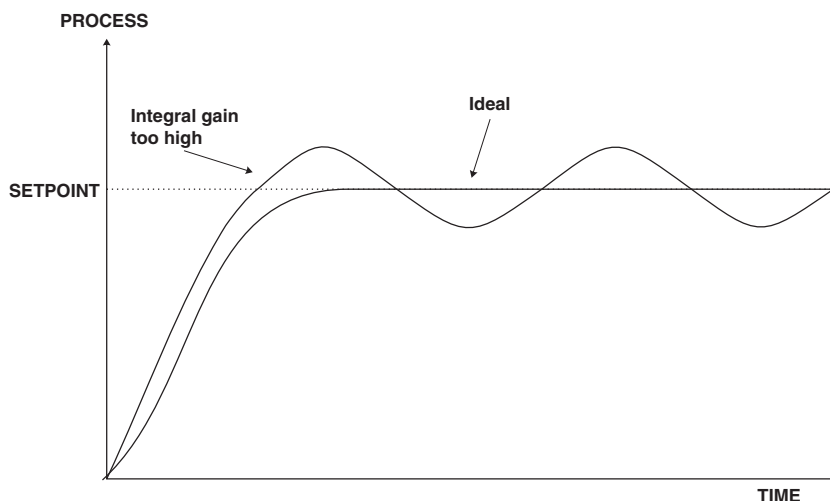
Display: **E: 9**

Range: **-32.767** to **32.767** number of display digits allowing

Default Value: **0.000**

The integral control action will attempt to correct any offset which the proportional control action is unable to correct (e.g. errors due to a changing load). When the integral gain is correctly adjusted

the control output is ramped up or down to maintain control by keeping the process variable at the same value as the control setpoint. An integral gain which is too large will cause a rapid response to any error but can also lead to overshooting and oscillation. An integral gain which is too small will slow the time taken to reach the setpoint. The optimum value chosen will depend on the lag time of the process and other control settings. Start with a low figure and increase until a satisfactory response time is reached. The integral gain figure has units of gain/minute. Setting a negative integral gain will reverse the integral control action. If introduction of an integral gain figure causes the error to increase i.e. the process value is moving further away from the setpoint then check the sign of the integral gain e.g. if it is negative change it to a positive value. Note that the sign of the integral gain value should be the same as the proportional gain value i.e. they should either both be positive or both be negative.



### 3.9 PI analog integral limit high

Display: **01 L.H**  
 Range: **0.0 to 100.0**  
 Default Value: **0.0**

The high limit sets the maximum control output for the integral term i.e. puts a high level limit to the integral control current or voltage output. The limit is used to reduce available output swing and hence limit the effect of integral control output build up which can cause overshoot and instability in the system. If the process value is not close to the setpoint value then the integral control will see a large error. Since integral control output increases with time, the longer an error is seen the more the integral control output will build up. Unless the output is limited then once the process reaches the setpoint the integral control output can be very large (e.g. 100%) causing the process value to overshoot the control setpoint. A setting which is too high will result in allowing the integral control output to cause overshooting. A setting which is too low will result in the integral control output being limited to an extent which means that the setpoint cannot be reached. Start with a low figure e.g. 10.0 and increase the value until a satisfactory response is reached. Maximum setting is 100.0 (100%). Having separate high and low limits is particularly useful if the process response is very one directional. For example in temperature control a heater may be used to give a fast response in heating a tank of liquid when the temperature falls below the setpoint. The heat of the liquid rises quickly but any overshoot will mean that the temperature is too high. The heater will be switched off but the tank of liquid will take a long time to cool to the setpoint level.

### 3.10 PI analog integral limit low

Display: **C I L L**  
Range: **0.0** to **100.0**  
Default Value: **0.0**

This function sets the minimum control output for the integral term value and works in the same manner as **C I L H** described above except that the setting controls the low swing.

### 3.11 PI analog setpoint access

Display: **F E C S P A C**  
Range: **o n** or **O F F**  
Default Value: **O F F**

This function determines whether the control setpoint function **C S E t** can be accessed via **F U N C** mode or whether entry via **C A L** mode is needed to access **C S E t**. If the operator is to have access to the **C S E t** function (via **F U N C** mode) then set the **F E C S P A C** function to **o n**. To make the access to the **C S E t** function more difficult (**C A L** mode) then set the **F E C S P A C** function to **O F F**. Note that in some models the control setpoint value can be reached and adjusted via the “easy access” mode (see “Explanation of functions” chapter in the main manual). The **A C C S** function must be set to **E A S Y** and the **F E C S P A C** function set to on to allow “easy access”. This feature could be useful if the setpoint is to be frequently changed. If no **F E C S P A C** is set to **o n** and **A C C S** is set to **E A S Y** but the easy access is not functioning then the “easy access” facility may not be available on that instrument.

### 3.12 Setting up the PI analog controller

1. Set the **F E C c t r l** function to **o n**.
2. Set the control setpoint **C S E t** to the required setting.
3. Set the proportional control span **C S P n** as required.
4. Set the proportional gain **C P g** to an arbitrary value e.g. 1.000.
5. Set the proportional offset **C P O** to 0.0 (0%).
6. Set the integral gain **C I g** to 0.000 (i.e. off).
7. Set the integral high and low limits to an arbitrary value e.g. 20.00.

Initialise the control system and monitor the control results. If the original settings causes process oscillations then gradually decrease the proportional gain until the oscillations decrease to an acceptable steady cycle. If the original settings do not cause process oscillations then gradually increase the proportional gain until a steady process cycling is observed. Once the steady state is achieved note the difference between the display value and the control setpoint value. Gradually increase or decrease the proportional offset value until the displayed value matches the control setpoint value. If process load changes occur then the proportional offset value may no longer be valid for offset free control. By introducing the integral action, setpoint offset caused by the process load changes will be minimised. Gradually increase the integral gain until the process begins to

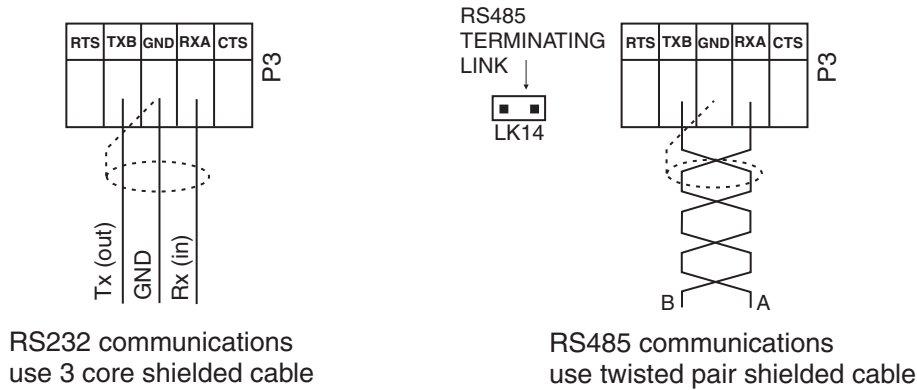
oscillate. Then reduce the integral gain slightly to regain the control with minimum oscillation. Alter the high and low integral limits to give the best regulation with minimum oscillation. Create a step change to the process conditions and observe the control results. It may be necessary to fine tune the settings to obtain optimum results. The table below summarises the effect of the main function settings.

Setup functions	Symptom	Solution
Proportional gain	Slow response	Increase proportional gain
	High overshoot or oscillation	Decrease proportional gain
Proportional offset	Process continually either above or below setpoint	Increase or decrease offset to compensate
Integral gain	Slow response	Increase integral gain
	Instability or oscillation	Decrease integral gain

## 4 Serial communication

This section deals with the optional serial communications output. The required circuitry will only be fitted if the instrument was ordered with this option.

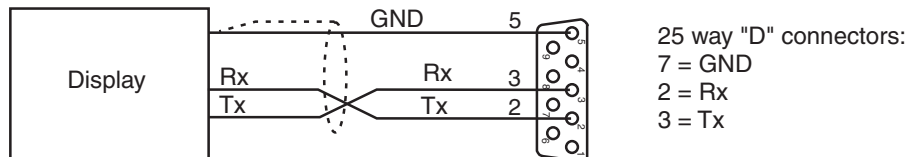
### 4.1 Serial electrical connections



Notes: When connecting using RS232 in most cases the Tx line at the display connects to the Rx line at the device it is communicating with and the Rx line at the display connects to the Tx line at the other device, see diagram below.

The RS485 terminating link is normally only required when long cable runs are used and communication difficulties are encountered due to data reflection. If multiple instruments are connected to the same RS485 chain then set the terminating link at the first and last instrument in the chain.

Standard PC 9 pin male "D" type RS232 serial port connector.  
Rear terminals (solder side) shown.



RS485 connection terminals may vary, check documentation when connecting.  
Terminal A is sometimes labeled "+" and terminal B is sometimes labeled "-"

### Serial commands

The serial commands **bAud**, **Prty**, **QPut** and **Addr** are described below. Refer also to the **SEFL** and **SEF: TYPE** functions in the main instruction manual.

### 4.2 Baud rate for optional serial communications

Display: **bAUD RATE**  
Range: **300.600.1200.2400.4800.9600.19.2** or **38.4**  
Default Value: **9600**

Set baud rate - seen only with serial output option. Select from **300.600.1200.2400.4800.9600.19.2** or **38.4** baud.

### 4.3 Parity for optional serial communications

Display: **Prty**  
Range: **NONE**, **EVEN** or **odd**  
Default Value: **NONE**

Set parity - seen only with serial output option. Select parity check to either **NONE**, **EVEN** or **odd**.

### 4.4 Output mode for optional serial communications

Display: **O.Put**  
Range: **di SP**, **Cont**, **POLL**, **A.buS** or **~.buS**  
Default Value: **Cont**

Set serial interface mode - seen only with serial output option. Allows user to select the serial interface operation as follows:

**di SP** - sends image data from the display without conversion to ASCII.

**Cont** - sends ASCII form of display data at a rate typically 90% of the sample rate.

**POLL** - controlled by computer or PLC as host. Host sends command via RS232/485 and instrument responds as requested.

**A.buS** - is a special communications mode used with Windows compatible optional PC download software. Refer to the user manual supplied with this optional software.

**~.buS** - Modbus RTU protocol.

Refer to section 5 for detailed description of the **di SP** and **Cont** options.

Refer to section 5.1 for detailed description of the **POLL** option.

Refer to section 6 for detailed description of the **~.buS** option.

### 4.5 Instrument address for optional serial communications

Display: **Addr**  
Range: **0** to **31**  
Default Value: **0**

Set unit address for polled (**POLL**) mode (**0** to **31**) - seen only with serial output option. Allows several units to operate on the same RS485 interface reporting on different areas etc. The host computer or PLC may poll each unit in turn supplying the appropriate address. The unit address ranges from 0 to 31 (DEC) but is offset by 32 (DEC) to avoid clashing with ASCII special function characters (such as <STX> and <CR>). Therefore 32 (DEC) or 20 (HEX) is address 0, 42 (DEC) or 2A (HEX) is address 10.

## 5 Serial operation

The operation mode for serial communication is set by the **G.PUL** function, the following choices are available:

**NONE** - No serial communication

**d. SP** - Image display mode.

In image display mode the display value is sent via RS232/RS485 as raw data in the format <ESC> IXYYYY where:

Where: <ESC> is the ESCAPE character (27 Dec, 1B Hex)  
I is the character "I" (73 Dec, 49 Hex)  
X is the number of image bytes in ASCII (31 to 38 Hex)  
YYYY is the raw, 8 bit display data.

This information is output every display update (approx. 4 times per second - depending upon baud rate). The number of image bytes sent depends on the number of display digits present. This mode is suitable only when the receiving unit is produced by the same manufacturer as the LD. The most common usage would be to provide a large digit display for wide area viewing which just mimics the smaller display on the measuring instrument. The large digit displays automatically detect the image mode data and display the correct value accordingly. The data is in seven segment display image i.e. Bit 0 is segment A, Bit 1 is segment B etc.

**cont** - Continuous mode.

In this mode the display value is continually sent via the RS232/485 interface in ASCII format with 8 data bits + 1 stop bit. Data will be updated at approximately 90% of the display sample rate. The format for this output is: <STX> XYYYY <CR>

Where: <STX> is start of text character (2 Dec, 02 Hex)  
X SPACE (32 Dec, 20 Hex) indicates a positive value or  
X "-" (45 Dec, 2D Hex) indicates a negative value.  
YYYY is the display value in ASCII.  
<CR> is a Carriage Return (13 Dec, 0D Hex).

e.g.: If the display is showing 123456 then the instrument will send "02 31 32 33 34 35 36 0D" (HEX) to the host.

**POLL** Host controlled transmit mode, see section 5.1.

**A.buS** Special communication mode for use with optional Windows compatible software, refer to the "Download Software User Guide" booklet supplied with this optional software.

**M.buS** Modbus RTU operation, see section 6

### 5.1 POLL mode commands

This mode requires a host computer. PLC or other device to poll the instrument to obtain display or other information or reset various setpoint parameters. Special communications software such as a terminal program is required when using **POLL** mode. Data is in ASCII format with 8 data bits + 1 stop bit. When polling the instrument it is essential that the command characters are sent with less than a 10mS delay between them. This normally means that each command line must be sent as a whole string e.g. a command such as <STX> PA <CR> is sent as one string



rather than  $\langle STX \rangle$  on one line followed by P etc. If testing using a “terminal” program or other software this is normally achieved by allocating a command string to a function key. Whenever the function key is operated the whole string is sent. The format used is ASCII (8 data bits + 1 stop bit) so, for instance, if address 1 is used then the string  $\langle STX \rangle PA \langle CR \rangle$  must be put into the terminal program as:  $\wedge BP! \wedge M$  where:

- $\wedge B$  is the ASCII character for  $\langle STX \rangle$  (2 Dec, 02 Hex)
- $P$  is the command line to transmit the primary display value (80 Dec. 50 Hex)
- $!$  is the ASCII character for address 1 (33 Dec of 21 Hex)
- $\wedge M$  is the ASCII character for  $\langle CR \rangle$  (13 Dec. 0D Hex)

Typical formats for the host command is as follows:-

- $\langle ST \rangle CA \langle CR \rangle$  (Standard read etc.)
- $\langle STX \rangle CA \langle CR \rangle N \langle CR \rangle YYYYYY$  (Set Value Command)

- Where:
- $\langle STX \rangle$  is Start of Text Character (2 Dec, 02 Hex,  $\wedge B$  ASCII)
  - $C$  is the command character (see following commands)
  - $A$  is the unit address (Range: 32 to 63 Dec, 20 to 3F Hex, “SPACE” to ? ASCII the address is offset by 32 Dec, 20 Hex)
  - $\langle CR \rangle$  is Carriage Return (13 Dec, 0D Hex,  $\wedge M$  ASCII)
  - $N$  is the setpoint number in ASCII e.g.: 1 for alarm 1 etc.
  - $X$  SPACE for positive and “-” for negative
  - $YYYY$  is the setpoint value in ASCII

**The POLL commands available and instrument responses are as follows:**

**Transmit primary display value:**  $\langle STX \rangle PA \langle CR \rangle$

e.g.  $\wedge BP! \wedge M$  using a terminal program (address 1). Instructs unit to return the primary display value. The primary value is the main reading. Format of returned data is:  $\langle ACK \rangle PAXYYYYY \langle CR \rangle$

- Where:
- $\langle ACK \rangle$  is Acknowledge (6 Dec, 06 Hex)
  - $P$  echo command received “P” (80 Dec, 50 Hex)
  - $A$  is the responding units address (offset by 32 Dec e.g. “!” is address 1)
  - $X$  SPACE for positive and “-” for negative
  - $YYYY$  is the display value in ASCII
  - $\langle CR \rangle$  is a Carriage Return (13 Dec, 0D Hex)

The number of display characters returned depends on the number of display digits present. If the decimal point is non zero then it will be sent in the appropriate place as “.” (46 Dec, 2E Hex).

**Transmit secondary display value:**  $\langle STX \rangle SA \langle CR \rangle$

e.g.  $\wedge BS! \wedge M$  using a terminal program (address 1). Instructs unit to return the secondary display value. If no remote input function is set then the secondary value is the same as the primary value. If a remote input is set for **H**, **Lo**, **H**, **Lo**, **P.HLd** or **d.HLd** then the value for the selected function will be returned e.g. if set to **H**, **Lo** the high value followed by the low value will be sent (separated by a comma). Format of returned data is:  $\langle ACK \rangle SAXYYYYY \langle CR \rangle$

Where: < ACK > is ASCII Acknowledge (6 Dec, 06 Hex)  
 S echo command received "S" (83 Dec, 53 Hex)  
 A is the responding units address (offset by 32 Dec e.g. "!" is address 1)  
 X SPACE for positive and "-" for negative  
 YYYY is the display value in ASCII  
 < CR > is a Carriage Return (13 Dec, 0D Hex)

**Reset special function value:** < STX > RA < CR >

e.g. ^BR!^M using a terminal program (address 1). Instructs unit to reset remote input1 value. Will reset the stored value for Peak Hold, Valley High and Valley Low or will operate the selected special function (tare, zero, batch or preset functions only). Format of returned data is < ACK > RA < CR >

Where: < ACK > is Acknowledge (6 Dec, 06 Hex)  
 R echo command received "R" (82 Dec, 52 Hex)  
 A is the responding units address (offset by 32 Dec e.g. "!" is address 1)  
 < CR > is a Carriage Return (13 Dec, 0D Hex)

**Read low alarm setpoint:** < STX > LA < CR > N < CR >

e.g. ^BN!^M2^M to read alarm 2 low setpoint using a terminal program (address 1). Instructs unit to return the low alarm setpoint value. Format of returned data is: < ACK > LANYYYYY < CR >, where:

< ACK > is Acknowledge (6 Dec, 06 Hex)  
 L echo command received "L" (76 Dec, 4C Hex)  
 A is the responding units address (offset by 32 Dec e.g. "!" is address 1)  
 N is the relay number in ASCII  
 X SPACE for positive and "-" for negative  
 YYYY is the setpoint value in ASCII  
 < CR > is a Carriage Return (13 Dec, 0D Hex)

**Read high alarm setpoint:** < STX > HA < CR > N < CR > // e.g. ^HN!^M2^M to read alarm 2 high setpoint using a terminal program (address 1). Instructs unit to return the high alarm setpoint value. Format of returned data is: < ACK > HANYYYYY < CR >, where:

Where: < ACK > is Acknowledge (6 Dec, 06 Hex)  
 H echo command received "H" (72 Dec, 48 Hex)  
 A is the responding units address (offset by 32 Dec e.g. "!" is address 1)  
 N is the relay number in ASCII  
 X SPACE for positive and "-" for negative  
 YYYY is the setpoint value in ASCII  
 < CR > is a Carriage Return (13 Dec, 0D Hex)

**Set low alarm setpoint:** < STX > lA < CR > N < CR > XYYYYY < CR >

e.g. ^lN!^M1^M1000^ to set alarm 1 low setpoint to 1000 using a terminal program (address 1). Instructs unit to set the low alarm setpoint value. Format of returned data is: < ACK > lANYYYYY < CR >

Where: < ACK > is Acknowledge (6 Dec, 06 Hex)  
 l echo command received "l" (108 Dec, 6C Hex)

*A* is the responding units address (offset by 32 Dec e.g. “!” is address 1)  
*N* is the relay number in ASCII  
*X* SPACE for positive and “-” for negative  
*YYYY* is the setpoint value in ASCII  
*< CR >* is a Carriage Return (13 Dec, 0D Hex)

**Set high alarm setpoint:** *< STX > hA < CR > N < CR > XYYYYY < CR >*

e.g.  $\wedge hN! \wedge M1 \wedge M5000 \wedge$  to set alarm 1 low setpoint to 5000 using a terminal program (address 1). Instructs unit to set the high alarm setpoint value. Format of returned data is: *< ACK > hANXYYYYY < CR >* where:

Where: *< ACK >* is Acknowledge (6 Dec, 06 Hex)  
*h* echo command received “h” (104 Dec, 68 Hex)  
*A* is the responding units address (offset by 32 Dec e.g. “!” is address 1)  
*N* is the relay number in ASCII  
*X* SPACE for positive and “-” for negative  
*YYYY* is the setpoint value in ASCII  
*< CR >* is a Carriage Return (13 Dec, 0D Hex)

**Transmit instrument model and software version:** *< STX > IA < CR >* e.g.  $\wedge BI! \wedge M$  using a terminal program (address 1). Instructs unit to return the instrument model and software version.

Format of returned data is: *< ACK > IACCX.X < CR >*

Where: *< ACK >* is Acknowledge (6 Dec, 06 Hex,  $\wedge F$  ASCII)  
*I* echo command received “I” (73 Dec, 49 Hex)  
*A* is the responding units address (offset by 32 Dec e.g. “!” is address 1)  
*CC* a 2 character identifier e.g. LC means loadcell input  
*X.X* is the software version number e.g. 4.6  
*< CR >* is a Carriage Return (13 Dec, 0D Hex,  $\wedge M$  ASCII)

**Invalid command:** If the command received from the host is invalid the unit will return the following: *< ACK > ?A < CR >*, where:

Where: *< ACK >* is Acknowledge (6 Dec, 06 Hex,  $\wedge F$  ASCII)  
*?* echo command received “?” (63 Dec, 3F Hex)  
*A* is the responding units address (offset by 32 Dec e.g. “!” is address 1)  
*< CR >* is a Carriage Return (13 Dec, 0D Hex,  $\wedge M$  ASCII)

## 6 Modbus operation

When using Modbus RTU communications the instrument must be set up electrically for RS232 or RS485 communications and the **Output** function must be set to **RS485**. The maximum recommended baud rate for Modbus operation is 9600. The following functions are available:

### Modbus Function 1 - Read coil status

Reads the ON/OFF status of the relay coils. Broadcast is not supported. Relay addresses are offset by 1 e.g. relay 1 is addressed as 0, relay 2 is addressed as 1 etc. Logic 1 = ON, Logic 0 = OFF. To read the coil status a query is sent to the instrument, the instrument then responds to the query. An example of a query to read coils 1 to 4 from the instrument at address 2 is given below.

Field name	Example(Hex)
Unit address	02
Function	01
Starting address Hi	00
Starting address Lo	00
Number of points Hi	00
Number of points Lo	04
Error check (LRC or CRC)	–

An example of a response is given below:

Field name	Example(Hex)
Unit address	02
Function	01
Byte count	01
Data (coils 4 to 1)	04
Error check (LRC or CRC)	–

The status of the relay coils is shown in the Data 04 (hex) or binary 0100. Relay 1 is indicated by the least significant binary bit. The status of the relays is therefore:

Relay 4 - OFF, Relay 3 - ON, Relay 2 - OFF, Relay 1 - OFF

### Function 3 - Read holding registers

This function reads the binary contents of the holding registers in the instrument being addressed. The value for this function is stored as a 32 bit two's complement number, 2 registers per channel are used. Note; a value of 1,000,000 represents a positive overrange and -200,000 a negative overrange. Registers 1 to 2 hold the display value, registers 3 to 4 the valley memory (lowest reading in memory), registers 5 to 6 the peak memory (highest reading in memory), registers 7 to 8 the display hold value. Registers 9 to 16 hold the alarm high values for relays 1 to 4. Note a value of 0X8000 means that the relay is set to OFF and has no high value. Registers 17 to 24 hold the alarm low values for relays 1 to 4. Note a value of 0X8000 means that the relay is set to OFF and has no low value. Register 25 represents the decimal point settings for the display. An example of a query to read holding registers 1 to 3 from the instrument at address 5 is given below.

<b>Field name</b>	<b>Example(Hex)</b>
Unit address	05
Function	03
Starting address Hi	00
Starting address Lo	00
Number of points Hi	00
Number of points Lo	03
Error check (LRC or CRC)	—

An example of a response is given below:

<b>Field name</b>	<b>Example(Hex)</b>
Unit address	05
Function	03
Byte count	06
Data Hi(register 1)	00
Data Lo(register 1)	33
Data Hi(register 2)	00
Data Lo(register 2)	25
Data Hi(register 3)	00
Data Lo(register 3)	17
Error check (LRC or CRC)	—

The contents of register 1 is 33 (hex) or 51 (decimal), register 2 is 25 (hex) or 37 (decimal), register 3 is 17(hex) or 23(decimal).

**Register table for LD displays using Modbus RTU function 3**

<b>Address</b>	<b>Register</b>	<b>Description</b>
0X00	1	Display value high word
0X01	2	Display value low word
0X02	3	Valley memory high word
0X03	4	Valley memory low word
0X04	5	Peak memory high word
0X05	6	Peak memory low word
0X06	7	Display hold high word
0X07	8	Display hold low word
0X08	9	Relay 1 high setpoint high word
0X09	10	Relay 1 high setpoint low word
0X0A	11	Relay 2 high setpoint high word
0X0B	12	Relay 2 high setpoint low word
0X0C	13	Relay 3 high setpoint high word
0X0D	14	Relay 3 high setpoint low word
0X0E	15	Relay 4 high setpoint high word
0X0F	16	Relay 4 high setpoint low word
0X10	17	Relay 1 low setpoint high word
0X11	18	Relay 1 low setpoint low word
0X12	19	Relay 2 low setpoint high word
0X13	20	Relay 2 low setpoint low word
0X14	21	Relay 3 low setpoint high word
0X15	22	Relay 3 low setpoint low word
0X17	23	Relay 4 low setpoint high word
0X17	24	Relay 4 low setpoint low word
0X18	25	Display decimal point

## 7 Data logger

The data logger is an optional addition to the instrument. This section applies only to instruments fitted with the data logger option. If the data logger is being used with the Windows compatible software provided then refer to the separate “Download Software User Guide” booklet.

### Operation of the data logger

The data logger memory will store the hours:mins:secs, day:month and year together with the display reading at the time of log update. The log update time may be set at the **LOGUPdt** function. If an input is overranged when logged then the overrange value ( - - - - ) will be logged for that channel for as long as the overrange value is present. Readings taken during power failure will not be logged. The log memory is set up in a circular format. Once the top of memory is reached the log data will overwrite the start of memory (overwriting the oldest record). The recording time available will vary depending on the memory size fitted and the update time selected. The table below shows maximum recording times.

Data is transmitted in comma separated format making it compatible with many commercially available databases/spreadsheets. Time information is downloaded in Julian time format which is again compatible with many databases/spreadsheets. The internal clock is battery backed. Downloaded log records are in the form of the time followed by the logged record for each channel at that time.

Downloaded information is transmitted via the serial output option board in RS232 or RS485 format, thus a serial output option must be fitted on all instruments with data logging software.

### Data logger Windows software

Data logger software compatible with Windows 95, 98, 2000, NT and XP is provided for use with the data logger (not tested and may not be compatible with Vista). A separate user booklet for the software is also provided. Consult this user manual for details of software setup. The data logger can also communicate using standard serial polling commands, these are listed under the heading “Serial Command Format” in this chapter.

### Datalogger table - maximum logging times (approximate)

Time between logs	32k memory days:hours	128k memory days:hours
10 seconds	0:13	2:04
20 seconds	1:02	4:08
30 seconds	1:15	6:12
1 minute	3:06	13:00
2 minutes	6:12	26:00
3 minutes	9:18	39:00
4 minutes	13:00	52:00
5 minutes	16:06	65:00
6 minutes	19:12	78:00
10 minutes	32:12	130:00
15 minutes	48:18	195:00
20 minutes	65:00	260:00
30 minutes	97:12	390:00
60 minutes	195:00	780:00

### Data logger polling functions

Usually data is downloaded using the Windows program supplied with the data logger but the data logger can be also polled via a PC etc. using the commands below. Functions which are used when

the data logger option is fitted are accessible only via **CAL** mode.

**LOG UPdt** (select log update time) Displays and sets the time period between each log sample. Available selections are: **0.10** (10 seconds), **0.20** (20 seconds), **0.30** (30 seconds), **1.00** (1 minute), **2.00** (2 minutes), **3.00** (3 minutes), **4.00** (4 minutes), **5.00** (5 minutes), **6.00** (6 minutes), **10.00** (10 minutes), **15.00** (15 minutes), **20.00** (20 minutes), **30.00** (30 minutes) or **60.00** (60 minutes).

Note: The data log memory (see **CLR LOG** below) must be cleared whenever the log update time is changed or the date and time is changed.

**CLR LOG** (clear data log memory) This function clears the data log memory, to clear the memory press then release **▲** and **▼** simultaneously, the display will show **Clr?** asking if you really want to clear the memory. If you wish to clear memory then press then release **▲** and **▼** simultaneously again. The log memory will then be cleared and the log period reset, the display will indicate **Prog LOG** to confirm this. Once the memory is cleared all previously logged records will be lost from the instruments memory, if the **Clr?** message is reached and it is not wished to clear the log memory then pressing and releasing either **F** or **P** will abort the function.

**SET ttc** (set time) Displays and sets the current time in hours and minutes (24 hour format HH.MM) e.g. set as **1720** for 5:20 pm.

**SET DATE** (set date) Displays and sets the current date in days and months (DD.MM format). The months will roll over automatically (up at the end of the month, down at the beginning of the month) as the day is scrolled up or down.

**SET YEAR** (set year) Displays and sets the current year (YYYY format). Valid years settings go up to 2037 (valid Julian time format years).

## Serial Command Format

Instruments using the data logger option are provided with extra software functions to the standard instrument. This section describes these extra functions.

### Initial Setup

Select the baud rate, parity and address as required. The serial output mode function (**0.Put**) must be set to **POLL** when using the data logger.

**Transmit Record Block:** <STX>DA<CR>D<CR>TTTTTTTTTT<CR>NNNN<CR>

Where: TTTTTTTTTT is the start time of the block (in Julian time format). NNNN is the number of records to be sent. Instructs the unit to send a block of logged data via the serial interface. The returned data format is: <ACK>DAD<CR> followed by NNNN records in the format:-

TTTTTTTTTT,S1111,S2222,S3333,S4444,S5555,S6666,S7777,S8888<CR> where:

TTTTTTTTTT is the start time for each record (in Julian time format). If TTTTTTTTTT (time in "Transmit Record Block" request) is sent as 0 then the records will start at the earliest time in log memory.

S is the sign (<SPACE> for positive values and "-" for negative.)

1111, 2222 etc. are the values for each channel.

Values will only be transmitted for active channels. Invalid readings from any channel will be received as the overrange value ( - - - - ) for that channel. If the start time requested is not present in the log then <ACK>DA?<CR> will be returned.



**Transmit All Logged Data:** <STX>DA<CR>A<CR>

Instructs the unit to transmit the entire data log. All log records since the last log memory reset will be sent to the host. The unit will respond with <ACK>DAA<CR> followed by all log record sent in the same format as above (Transmit Record Block)

**Transmit System Time:** <STX>DA<CR>T<CR>

Instructs the instrument to transmit the current time in Julian time format as follows:-  
<ACK>DAT TTTTTTTTTT<CR>

**Transmit the Log Start Time:** <STX>DA<CR>S<CR>

Instructs the instrument to transmit the log start time i.e. the time stamp on the first record in the log. Note that if the memory has “wrapped around”, i.e. has started to overwrite existing logged records, that the log start time will not be the original time the log started (since this time stamp and associated log record has been overwritten). The returned data format is: <ACK>DAS TTTTTTTTTT<CR>

**Transmit the Log Update Time:** <STX>DA<CR>U<CR>

Returns the current log update time as set in the log memory. The returned time may be different to the **DLRY** time if there has been no log reset since the **DLRY** function was changed. The returned data format is: <ACK>DAU NNNN<CR>, where NNNN is the update time in seconds.

**Transmit the Log Memory Size:** <STX>DA<CR>M<CR> Returns the size of the log memory in records. The returned data format is: <ACK>DAM NNNN<CR>, where NNNN is the number of records for that memory size e.g. an 8K memory will return 508.

**Set the System Time:** <STX>DA<CR>t<CR>TTTTTTTTTT<CR>

Set the instrument system clock to Julian time TTTTTTTTTT. If the command is successful then <ACK>DA<CR> will be returned. If the Julian time is invalid then <ACK>DA?<CR> will be returned.

**Set the Log Update Time:** <STX>DA<CR>u<CR>NNNN<CR>

Set the log update time to NNNN seconds. Note that the new time will not apply until a log reset is performed. If the command is successful then <ACK>DAu<CR> will be returned. If the update time is invalid then <ACK>DA?<CR> will be returned. Valid times are as shown in the **DLRY** function explanation.

**Reset the Log Memory:** <STX>DA<CR>R<CR>RESET<CR>

This command will reset the log memory. This will erase all current records and reset the log update time if it has changed. As this will result in a loss of data the command must be sent exactly as it appears or the memory will not be reset. If the command is successful then <ACK>DAR<CR> will be returned to indicate that the memory has been reset. If the command is invalid then <ACK>DA?<CR> will be returned.