

# CR - CONTROL RELAY

PC-1100-x01y: SUPPORTED	PC-1100-x05y: SUPPORTED
PC-1100-x02y: SUPPORTED	PC-1200-x02y: SUPPORTED
PC-1100-x03y: SUPPORTED	PC-1200-x04y: SUPPORTED

## DESCRIPTION

Control Relay (CR) coils control contacts and output circuits. Each coil controls all contacts with the same reference number as the coil. Output coils can control output circuits. Logic coils are for internal logic and control contacts. The ranges for the reference numbers for these coils depend on the model of programmable controller being used as shown in Table 1. CR function symbology is shown in Figure 1.

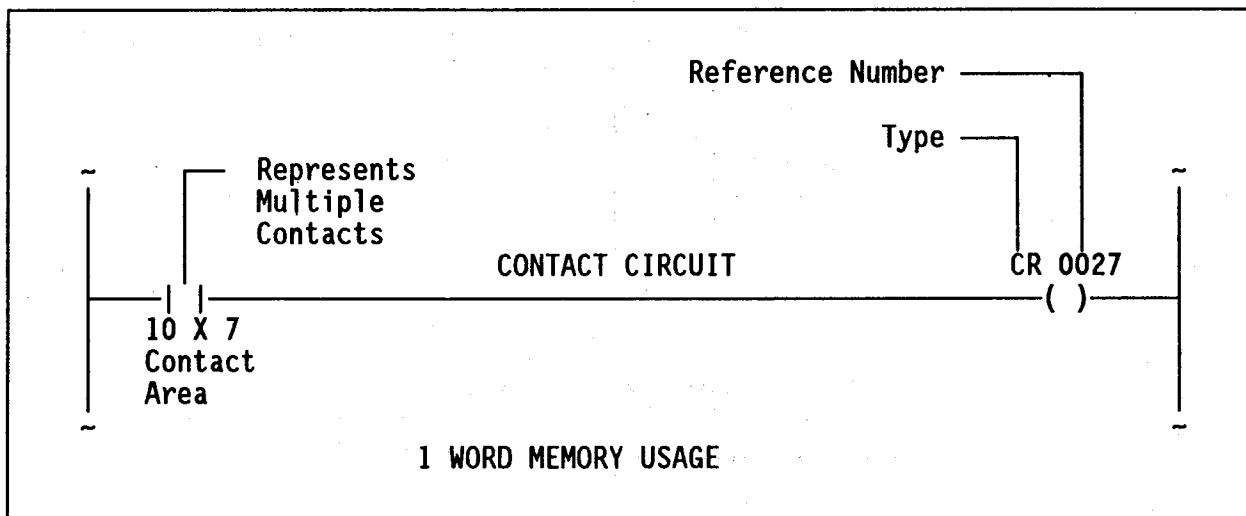
**TABLE 1. REFERENCE NUMBER RANGES FOR COIL**

Model(s)	Output	Internal Logic	Total CR	Battery Status
PC-1100	64	192	256	CR0128
PC-1200/1250 <sup>1</sup>	64/128/256	960/896/768	1024	CR1024

<sup>1</sup> Number of output and logic coils for PC-1200/1250 depends on specific model. The figures shown are for the PC-1200-xyy0, PC-1200-xyy1/2/3, and PC-1250, respectively. Refer to Section 4 for details.

### Note

When necessary, output coils that do not operate circuits operate as logic coils. It is wise to allow a gap in the numbering between output coils and logic coils to accommodate program expansion.

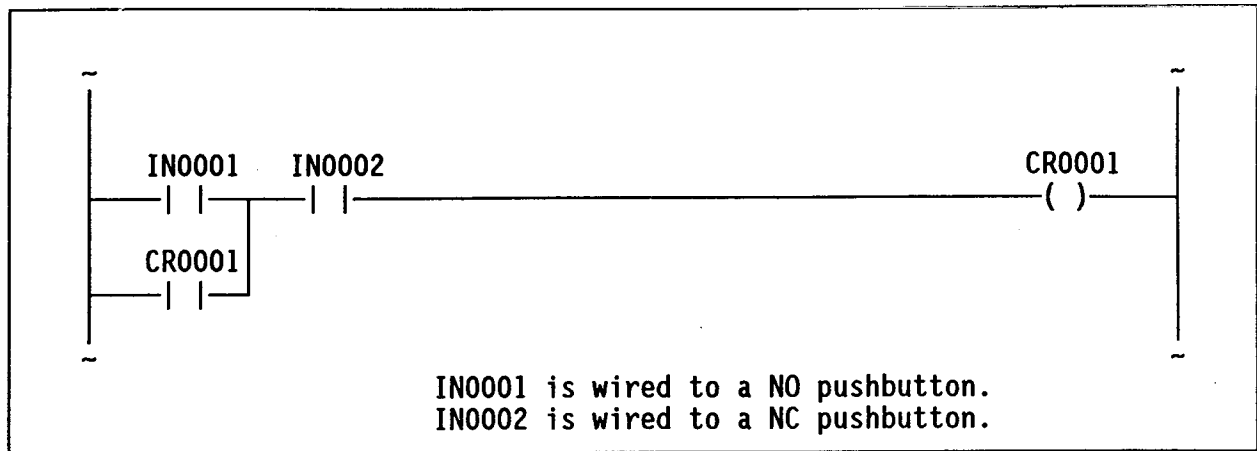


**Figure 1. Control Relay (CR)**

Each CR coil responds directly to its contact circuit and/or output module. When the contact circuit conducts, the coil is energized. When the contact circuit does not conduct, the coil is de-energized. When the coil is energized, normally-open (NO) contacts are closed, and normally-closed (NC) contacts are open. When the coil is de-energized, NO contacts are open and NC contacts are closed. All CR coils that are not forced are cleared during power-up or when the key switch changes from STOP to RUN.

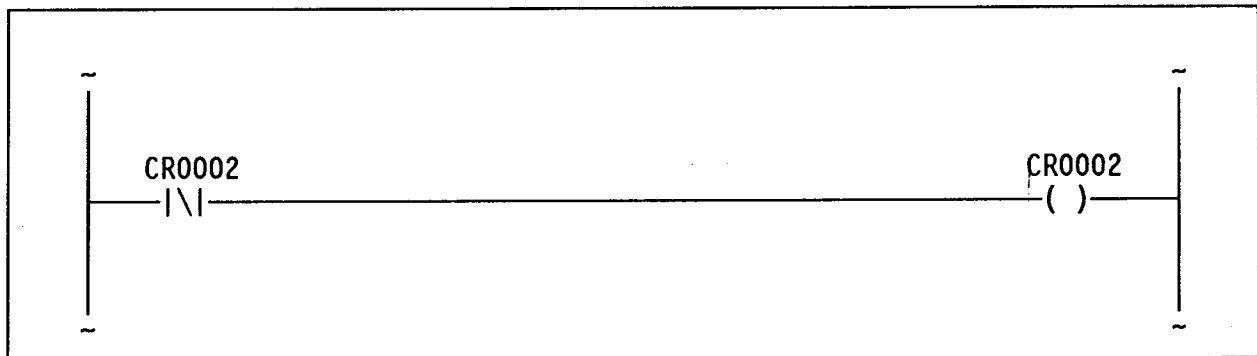
## APPLICATIONS

Figure 2 shows a control relay start/stop circuit application. As shown, when pushbutton IN0001 is pressed, CR0001 energizes through the NC contact of IN0002. This causes the CR0001 NO contact to close, holding Coil CR0001 energized until IN0002 is pressed to open the circuit.



**Figure 2. Start/Stop Circuit**

The oscillator circuit application of a control relay is shown in Figure 3. Upon the initial scan, the programmable controller detects the NC contacts of CR0002, energizing the CR0002 coil on the next scan. The NC contacts of CR0002 to subsequently de-energize. In this way, the CR0002 coil is energized every other scan.



**Figure 3. Oscillator Circuit**

# CR

Figure 4 shows the CR dummy coil circuit application. As shown, CR0003 is always OFF, unless forced ON. This configuration allows the CR0003 contacts to be used whenever a circuit is to be always open (NO contact) or always closed (NC contact). Not programming CR0003 has the same effect as shown in Figure 4; the CR0003 contacts can be used for always ON or always OFF circuits. However, when not programmed, the state of CR0003 is not documented, and it is not immediately apparent that the coil is used as a dummy circuit. Unless programmed as shown, CR0003 could be forced ON, staying ON even with a force deleted. When programmed, the coil turns OFF when the force command is removed.

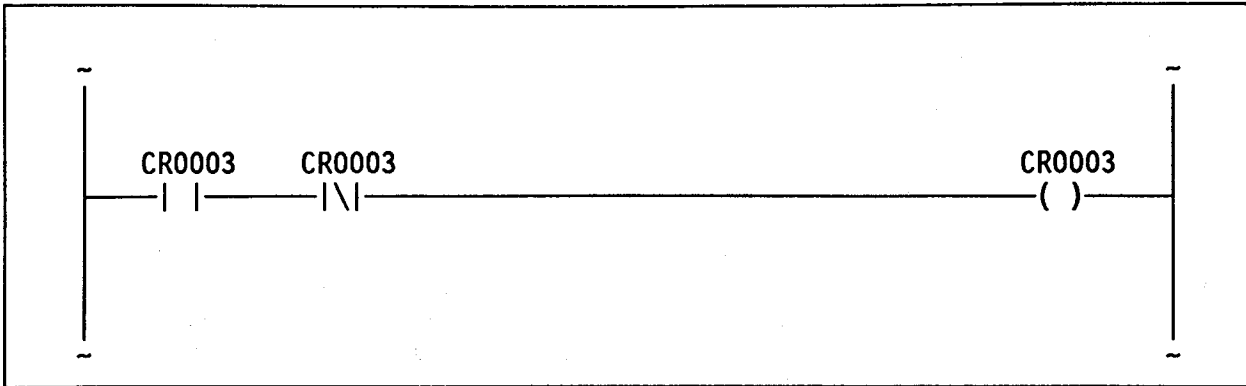


Figure 4. Dummy Circuit

Figure 5 shows a method of resetting the programmable controller when the key switch has been changed or after power-up. When placed at the end (last rung) of a program, this method provides for scan reset. The normally-closed contact conducts during the first scan only. The normally-open contact conducts thereafter.

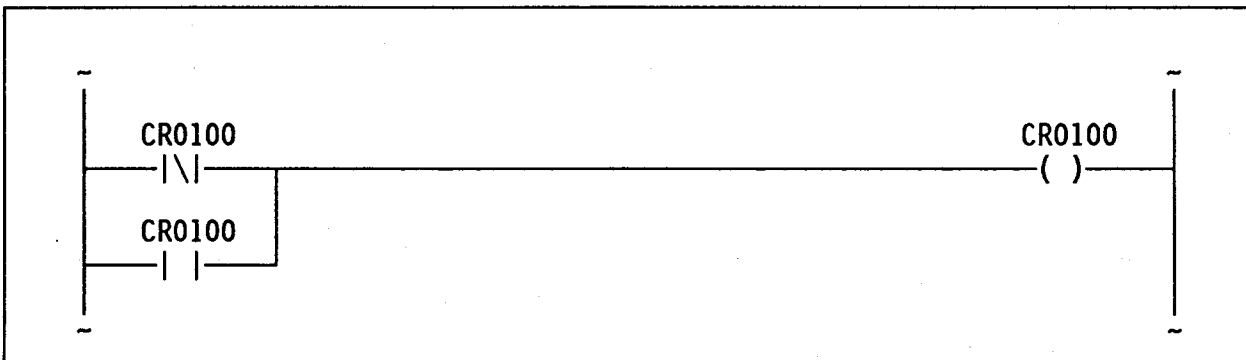


Figure 5. First Scan Reset

## DV - DIVIDE

Modified for PC-1200

PC-1100-x01y: NOT SUPPORTED	PC-1100-x05y: SUPPORTED
PC-1100-x02y: SUPPORTED	PC-1200-x02y: SUPPORTED
PC-1100-x03y: SUPPORTED	PC-1200-x04y: SUPPORTED

### DESCRIPTION

The Divide (DV) function divides one integer number (the dividend) by another (the divisor), resulting in an integer result (quotient) and remainder. DV function symbology is shown in Figure 1. Operand 1 is divided by Operand 2 when the enable circuit changes from non-conducting to conducting.

Operand 1 (the dividend) is held in a pair of input, output, or holding registers. These registers are specified by a single reference label which automatically designates itself (as the most-significant portion of the value) and the next register of the specified type (as the least-significant portion of the value).

Operand 2 (the divisor) is held in a single input, output, or holding register, or is programmed as a constant.

The division result is placed in a consecutive pair of output or holding registers. The specified Destination register contains the quotient; the next register of the specified type contains the remainder.

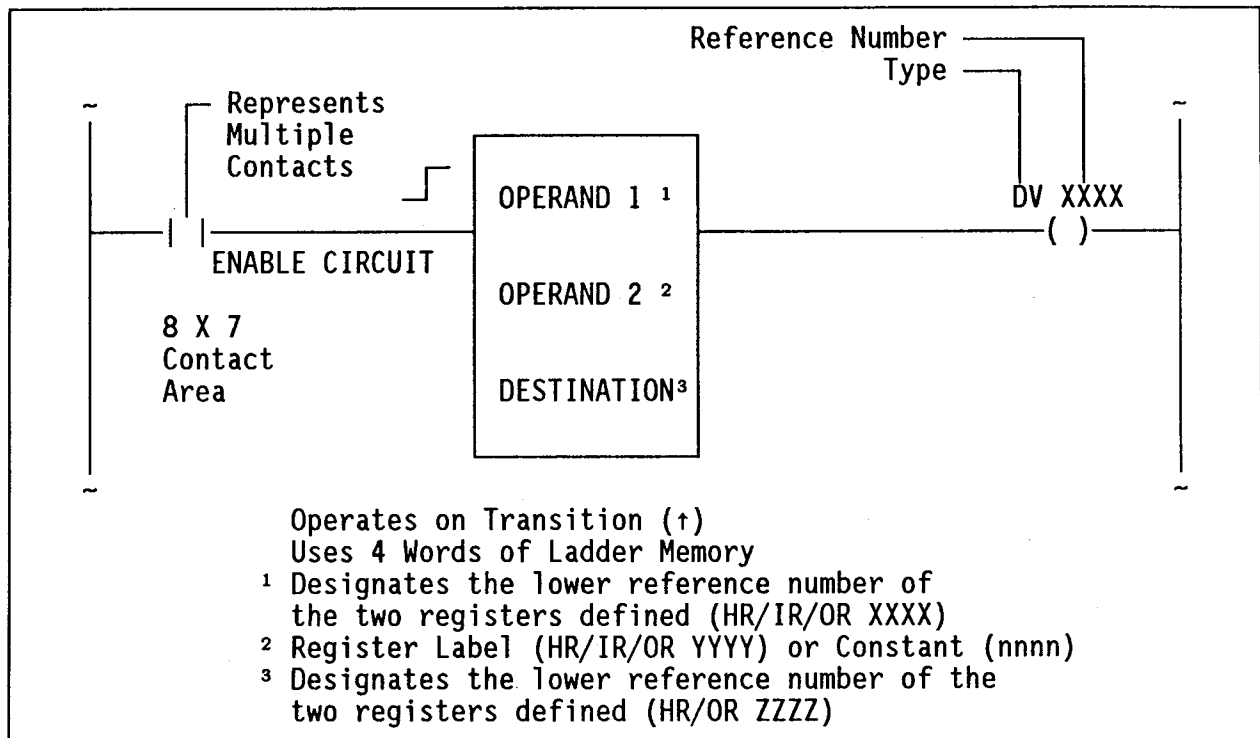


Figure 1. Division (DV)

## DV

For example, if IR0001 is specified for Operand 1, HR0001 is specified for Operand 1, and OR0001 is specified for Operand 3, the following assignments will be made:

IR0001 - Most significant word of dividend

IR0002 - Least significant word of dividend

HR0001 - Divisor

OR0001 - Quotient

OR0002 - Remainder

In general, the most significant word of the dividend value can be interpreted as "ten-thousands". For instance, if the value in the first register of Operand 1 is 9999 and the value in the second register of Operand 1 is 0001, then the dividend is calculated as follows:

(Most-significant word X 10,000) + (Least-significant word) =

(9999 X 10,000) + 1 = 99,990,001

All three of the operands of this function are subject to certain size restrictions. These limits are summarized below for the PC-1100 and for the PC-1200.

### Operand Limits for PC-1100

In the PC-1100, the primary limiting factor for the DV function is the quotient, which cannot exceed 9999. The dividend (Operand 1) can be any number up to 99,999,999. The divisor can be any number up to 65,535.

### Caution

**The maximum values for the dividend and divisor are valid only when the maximum for the quotient is not exceeded.**

Two examples are shown below:

#### A. VALID

Operand 1: 99999999  
divided by  
Operand 2: 65535  
equals  
Operand 3: 1525, Remainder 59124

#### B. NOT VALID

Operand 1: 30000  
divided by  
Operand 2: 2  
equals  
Operand 3: 15000, Remainder 0

Example A is valid because the quotient and all other operands are within limits. Example B is not valid, even though the values of the other operands are lower, because the quotient exceeds 9999.

### Operand Limits for PC-1200

For the PC-1200, the divisor, result, and quotient can be any number up to 65,535. The dividend (Operand 1) can be any number up to 655,415,535. Note that each of the registers that make up Operand 1 are limited to 65,535. As stated previously, the first register of Operand 1 is interpreted as "ten-thousands". Thus, the overall limit for Operand 1 is derived as shown below:

Specified Register:	65535----
Next Register:	----65535
Total:	<u>655415535</u>

### General Restrictions (PC-1100 and -1200)

Although the limits on the operands and the result are stated above in decimal, the division function is performed using binary numbers. Therefore, the register values for Operand 1 and Operand 2 must be stored in binary form. If a number is originally in Binary-Coded-Decimal (BCD) form, it may be converted to binary form by using the Decimal-to-Binary (DB) function. If the result is needed in BCD form, the Binary-to-Decimal (BD) function is used.

#### Note

Both operands must be in binary form.

The DV coil energizes if the divisor (Operand 2) equals 0, or if the quotient (Operand 3) is greater than the maximum (stated above). The coil energizes to indicate that an error has been detected. If there is an error, the destination register is not updated and contains the same value as before the function was enabled.

## SPECIFICATIONS

### OPERAND 1

Operand 1 is the dividend contained in a consecutive pair of registers. The first register contains the most-significant digits; the second register contains the least-significant digits. This value is held in a specified pair of Holding Registers (HR), Input Registers (IR), or Output Registers (OR).

### OPERAND 2

Operand 2 is the divisor. This value may be a constant or may be held in a specified Holding Register (HR), Input Register (IR), or Output Register (OR).

## DV

### DESTINATION

The destination is a consecutive pair of registers. The first register contains the result; the second register contains the remainder. The destination is a specified pair of Holding Registers (HRs) or Output Registers (ORs).

### DV TRUTH TABLE

See Table 1.

TABLE 1. DV TRUTH TABLE

Enable Circuit	Result
0	None. The DV coil de-energizes.
↑	Operand 1 is divided by Operand 2. The result is placed in the destination register. The coil energizes when the circuit is enabled and either Operand 2 is zero or the result is greater than the maximum; the destination retains the last valid computed value.
1	None. The coil retains the state caused by the transition.

### APPLICATIONS

It is desirable to scale inputs from an 8-bit analog input module (range of binary input is 0 - 255) to the appropriate value between 0 and 5000. Use the following formula to perform the desired scaling:

$$\frac{(\text{received input value}) \times (\text{desired maximum value})}{(\text{maximum possible received input value})} = \text{scaling factor}$$

In Figure 2, this formula is implemented. Note that HR0011 reads the value from an A/D module, HR0012 (or a constant value) contains the desired maximum output, HR0015 (or constant value) contains the highest value the input can achieve, and the scaled result is in HR0016. HR0013, HR0014, and HR0017 contain intermediate results and a remainder.

#### Note

The maximum value of the operands is restricted, as described previously.

Figure 3 shows a 16-bit word divided into two eight-bit words.

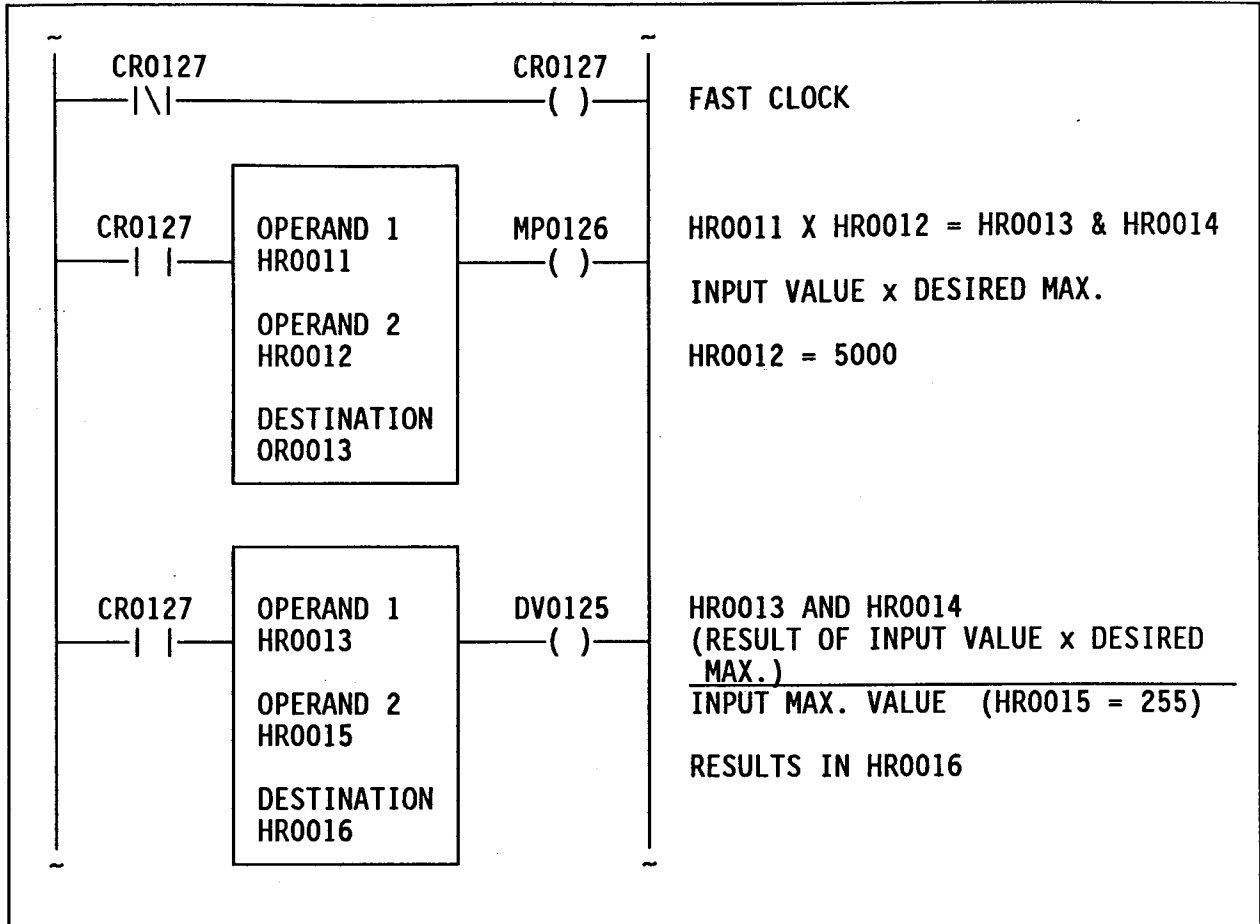


Figure 2. DV Scale Factoring

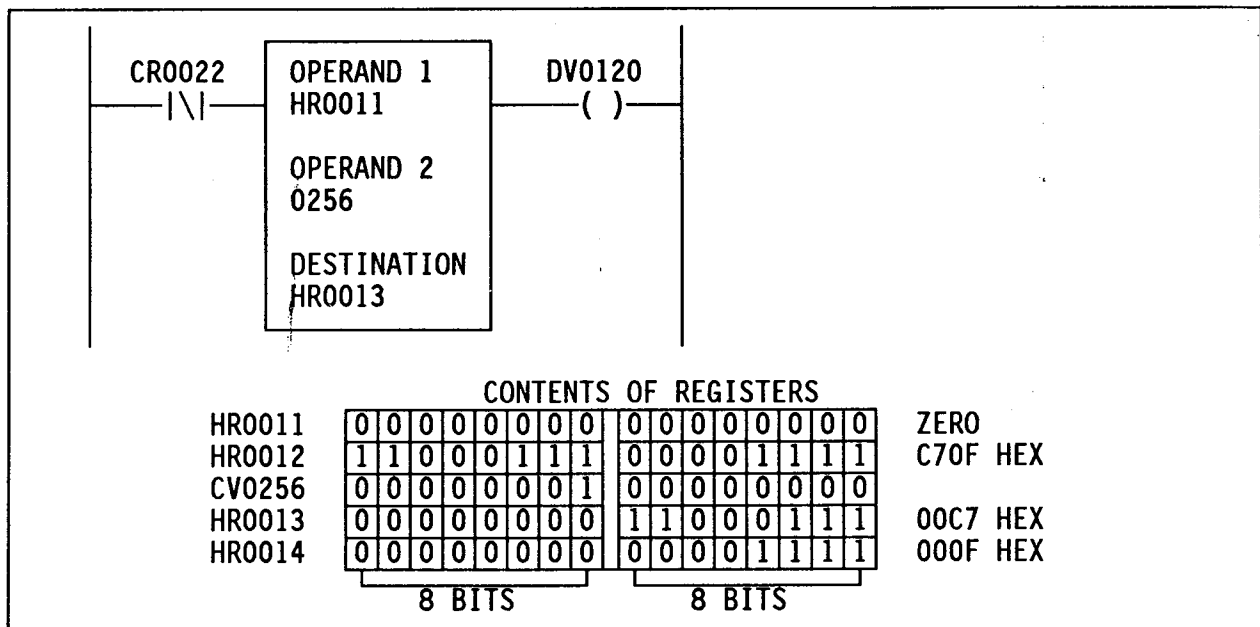


Figure 3. 16-Bit Word Divided into Two, Eight-Bit Words



# EQ/GE - COMPARISON

PC-1100-x01y: SUPPORTED	PC-1100-x05y: SUPPORTED
PC-1100-x02y: SUPPORTED	PC-1200-x02y: SUPPORTED
PC-1100-x03y: SUPPORTED	PC-1200-x04y: SUPPORTED

## DESCRIPTION

There are two Comparison functions: Equal To (EQ) and Greater Than Or Equal To (GE). The EQ/GE functions operate a coil by comparing two four-digit decimal numbers. EQ/GE function symbology is shown in Figure 1.

The range of this function is 0 - 65535 when entered as the contents of a register. The range is 1 - 9999 when entered as a constant value in Operand 2 only. Numbers compared are assumed to be decimal. Binary contents are compared if one number is represented as BCD. Both operands must be BCD and the limit is 9999.

The two numbers, Operand 1 and Operand 2, are compared when the enable circuit conducts. Operand 1 comes from an input, holding, or output register. Operand 2 comes from one of these registers, or is programmed as a constant value.

Any comparison ( $<$ ,  $>$ ,  $\neq$ ,  $=$ ,  $\leq$ ,  $\geq$ ) is made by carefully choosing the type of contacts used or by combining the contacts from both the EQ and GE functions. For example, if the comparison function is A equals B ( $A = B$ ) and if the enable circuit conducts, the normally-closed (NC) contact is closed when A does not equal B. Similarly, the NC contacts from a GE coil indicate when A is less than B ( $A < B$ ). A comparison of A greater than B ( $A > B$ ) can be generated by the series combination of a normally-open (NO) GE contact and an NC EQ

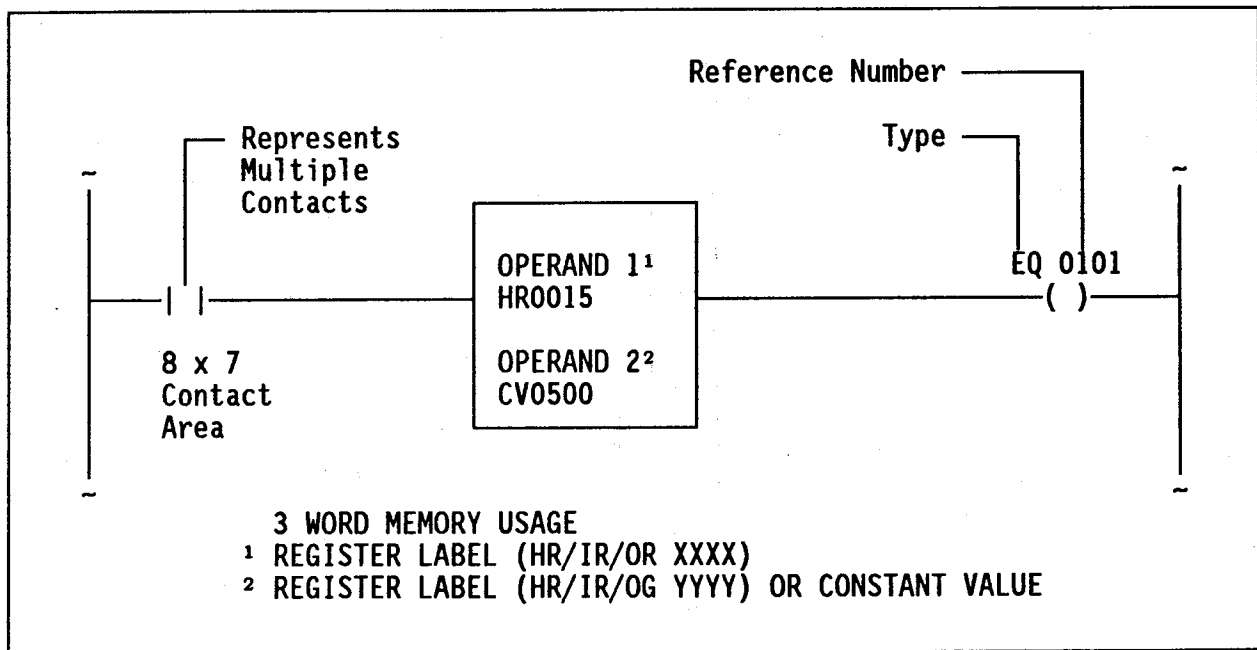
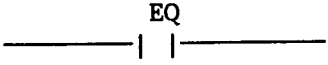
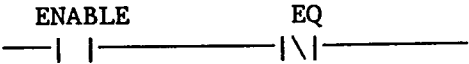
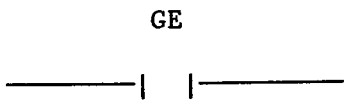
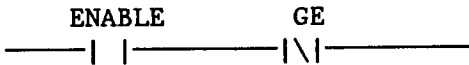

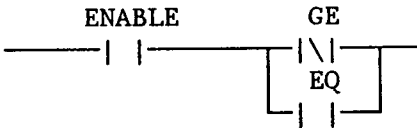


Figure 1. Equal To (EQ)/Greater Than Or Equal To (GE)

contact. A comparison of A less than or equal to B ( $A \leq B$ ) can be generated by the parallel combination of an NC GE contact and an NO EQ contact. When comparison functions are used together in this manner, identical contact circuits should be used for both functions. When normally-closed contacts are used, the contact of the enable circuit should be added in series.

Table 1 lists the six possible comparisons and the condition of the circuit when the equation is true.

TABLE 1. COMPARISON FUNCTIONS

Function	Equation	Circuit (Conducts When Equation is True)
Equal (EQ)	$A = B^1$	
Not Equal	$A \neq B^1$	
Greater Than or Equal To (GE)	$A \geq B^1$	
Less Than	$A < B^1$	
Greater Than	$A > B^1$	
Less Than or Equal To	$A \leq B^1$	

<sup>1</sup>Operand 1 contains A; Operand 2 contains B

## EQ

When the enable circuit conducts and Operand 1 equals Operand 2, the EQ coil energizes. If Operand 1 and 2 are not equal, or if the enable circuit does not conduct, the coil de-energizes.

## GE

When the enable circuit conducts and Operand 1 is greater than or equal to Operand 2, the GE coil energizes. If Operand 1 is less than Operand 2, or if the enable circuit does not conduct, the coil de-energizes.

## **EQ/GE**

### **SPECIFICATIONS**

#### **ENABLE CIRCUIT**

When the enable circuit conducts, Operand 1 is continuously compared to Operand 2. When the circuit does not conduct, the function disables and the coil de-energizes.

#### **OPERAND 1**

Operand 1 is the number to be compared (A). This value is held in a specified register:

- Holding Register (HR)
- Input Register (IR)
- Output Register (OR)

#### **OPERAND 2**

Operand 2 (B) is the number to which Operand 1 (A) is compared. This value is a constant (0001 through 9999) or is held in a specified register:

- Holding Register (HR)
- Input Register (IR)
- Output Register (OR)

#### **COIL**

The EQ coil energizes when the enable circuit is conducting and Operand 1 equals Operand 2.

The GE coil energizes when the enable circuit is conducting and Operand 1 is greater than or equal to Operand 2.

Both functions de-energize at all other times.

APPLICATIONS

The GE/EQ functions allow monitoring of an analog input within a range of values. In the Add (AD)/Subtract (SB) functions, a method of setting a range around a set point is discussed. Using the limit registers established in that example, HR0003 holds the upper limit, and HR0005 holds the lower limit, as shown in Figure 2.

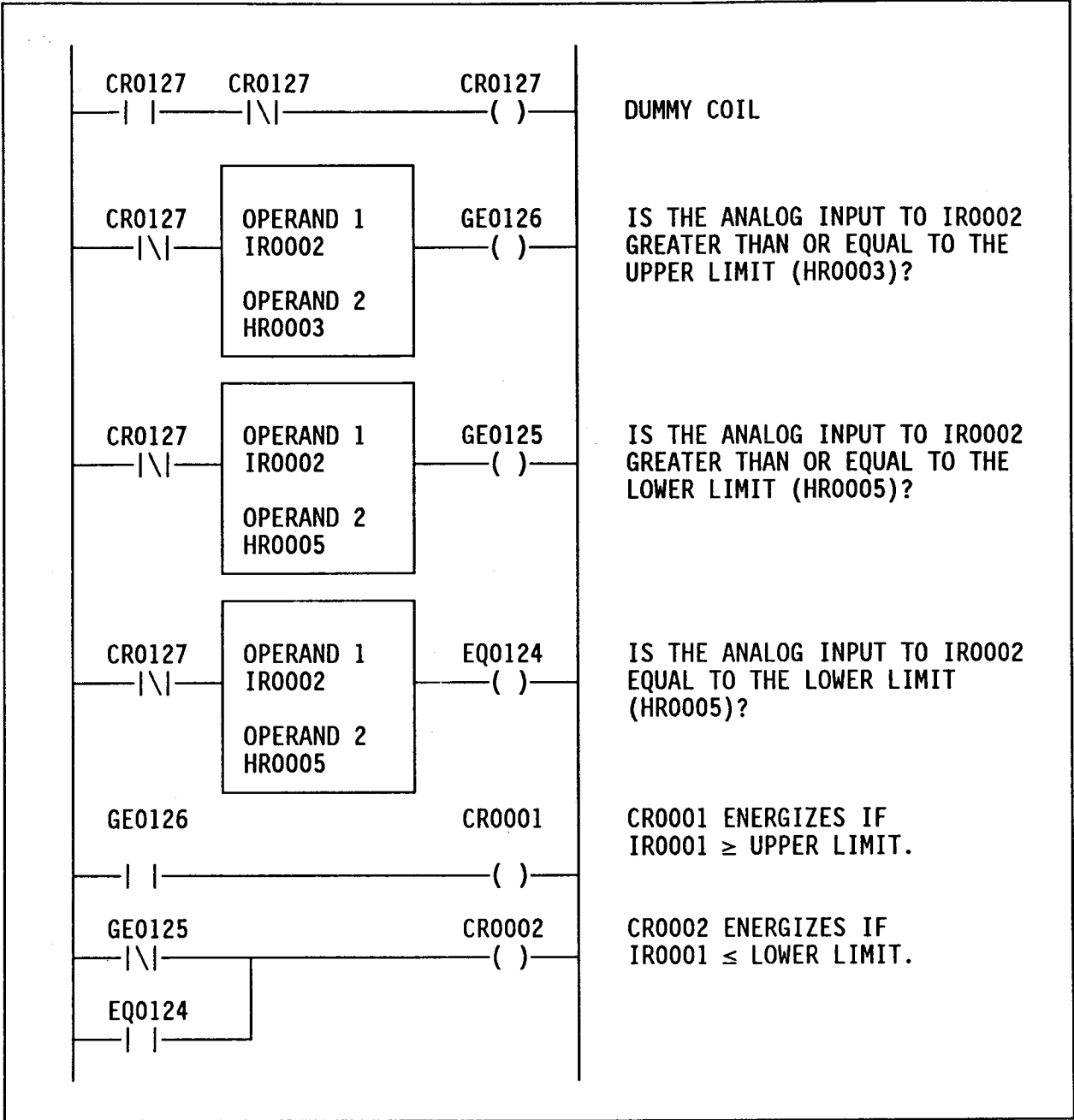


Figure 2. Limit Comparisons

# FI/FO-FIFO STACK

Modified for PC-1200

PC-1100-x01y: NOT SUPPORTED	PC-1100-x05y: SUPPORTED
PC-1100-x02y: SUPPORTED	PC-1200-x02y: SUPPORTED
PC-1100-x03y: SUPPORTED	PC-1200-x04y: SUPPORTED

## DESCRIPTION

The FIFO Stack function is a combination of two functions: the First in Stack (FI) function, and the First Out Fetch (FO) function. In FIFO Stack, the first data entered is the first data retrieved. FI function symbology is shown in Figure 1; FO function symbology is shown in Figure 2.

### Note

The FI and FO functions are designed to be used together.

Figure 3 uses a card playing analogy to demonstrate the FIFO Stack concept. Individual cards are stacked on a table; then, they are dealt from the bottom of the stack. When Card No. 1 is removed, Card No. 2 occupies the bottom position.

## OP CODE

Op Code 81 and 85 defines the Literal (LT) as either a FI or FO function. Whether or not the Literal function should be used depends upon the capability of your program loader. It is recommended that the mnemonic function be used whenever possible. Refer to the Introduction in this Section and the LT function description.

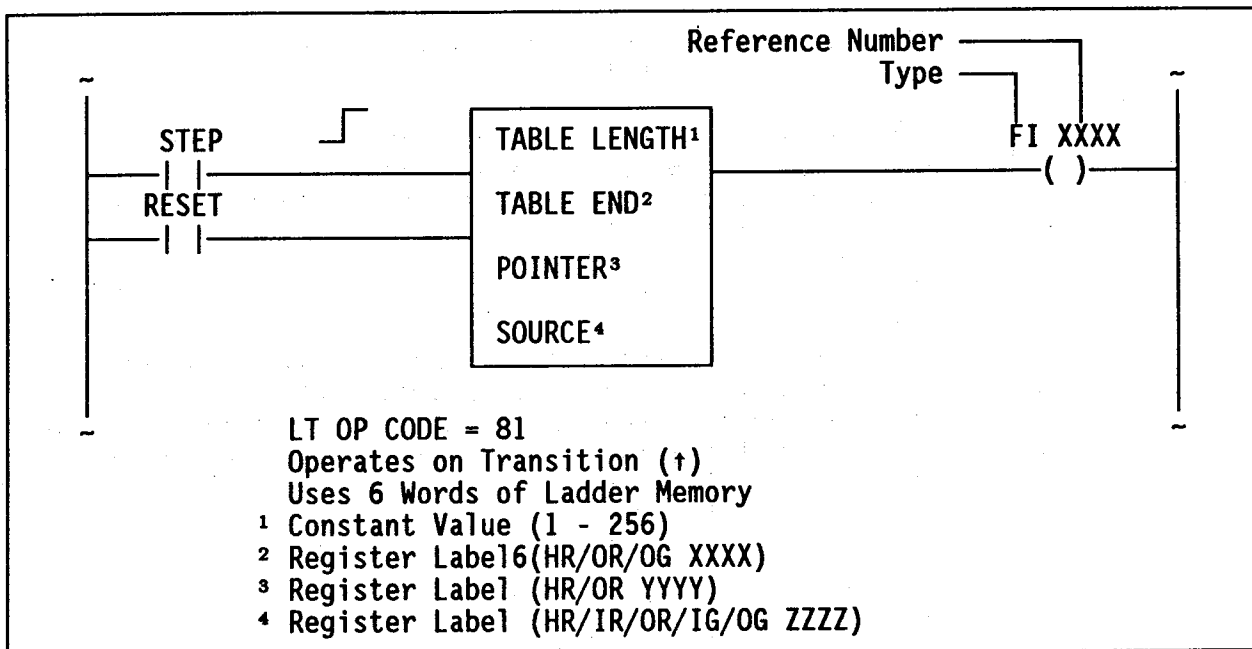


Figure 1. FI Function

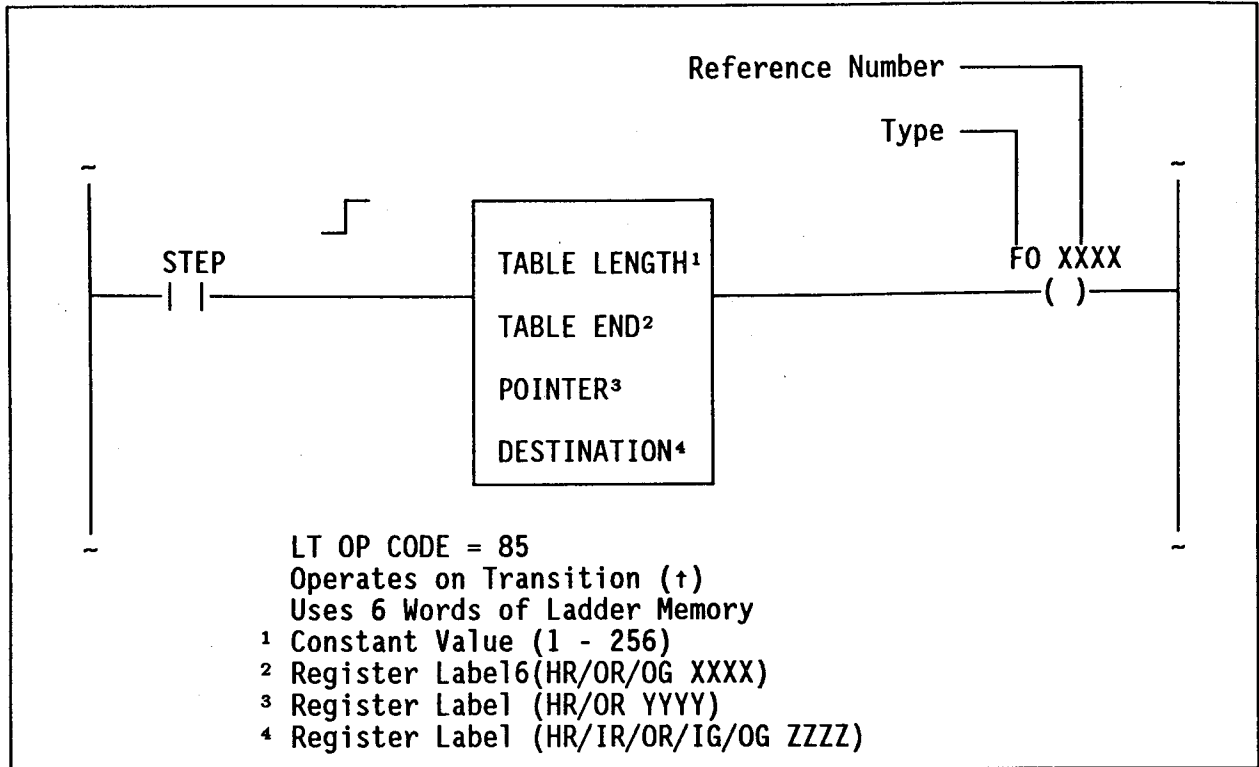


Figure 2. FO Function

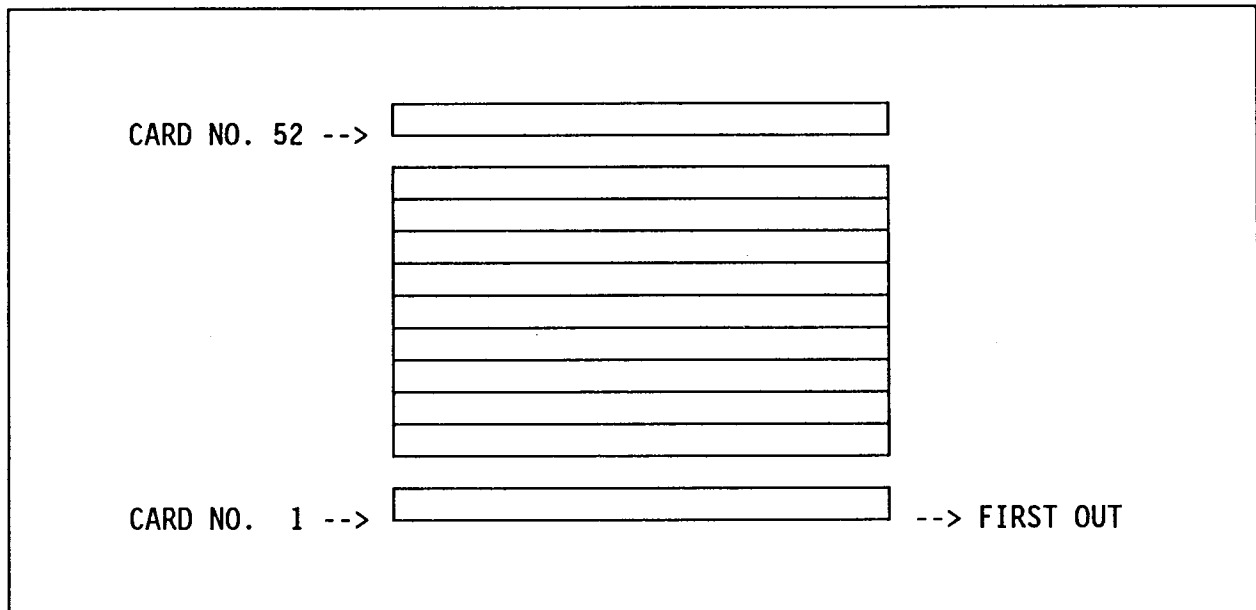


Figure 3. FIFO Stack Concept

# FI/FO

## SPECIFICATIONS

### OPERAND 1 - TABLE LENGTH

The table length is a constant value that defines the size of the FIFO Stack (i.e., number of registers). Both functions should be programmed with the same table length. The size of the FIFO Stack is limited, as specified in Table 1, with a maximum of 256.

**TABLE 1. TABLE LENGTH AND TABLE END LIMITS**

Type	PC-1100	PC-1200-1020/1040	PC-1200-1041/1041/1042	PC-1250
HR	$\leq 1792$ <sup>1</sup>	$\leq 1792$	$\leq 1792$	$\leq 1792$
OR	$\leq 8$	$\leq 32$	$\leq 64$	$\leq 128$
OG	$\leq 8$	$\leq 32$	$\leq 64$	$\leq 128$

<sup>1</sup> For the PC-1100, the maximum number of holding registers depends on memory size, as described in Section 4.

#### Note

The highest Holding Register reference number acceptable is dependent on the memory and user program size.

### OPERAND 2 - TABLE END

The table end defines the type and number of the last register in the FIFO Stack, which is subject to the limits in Table 1. Both functions should be programmed with the same table end.

### OPERAND 3 - POINTER

The pointer is a register that contains the number of items in the FIFO Stack. This register must be the same in both functions:

- Holding Register (HR)
- Output Register (OR)

**OPERAND 4 (FI) - SOURCE**

The source in the FI function is the location from which data is moved to the the Stack. This location is a specified register or group:

- Holding Register (HR)
- Input Register (IR)
- Output Register (OR)
- Input Group (IG)
- Output Group (OG)

**OPERAND 4 (FO) - DESTINATION**

The destination is used in the FO function to determine the location to which data is moved from the Stack. This location is a specified register or group:

- Holding Register (HR)
- Output Register (OR)
- Output Group (OG)

**FI TRUTH TABLE**

See Table 2.

**TABLE 2. FI TRUTH TABLE**

Step	Reset	Result
X	0	The coil de-energizes. The pointer sets to zero.
↑	1	The source moves to the position in the stack designated by the pointer. After the move, the pointer increments by 1. See explanation below for PC-1200.
X	1	The coil energizes when the pointer value equals the stack size, and de-energizes when the pointer value is less than the stack size.
X = Don't care		



## FI/FO

The stack grows from the highest numbered holding register (stack end) toward the lower numbered registers. The pointer always points to the next available stack location. A pointer value of zero points to the first stack location (stack end). A pointer value of N-1, where N is the specified stack length, points to the last available stack location. A pointer value equal to the stack length implies a full stack.

### FO TRUTH TABLE

See Table 3.

TABLE 3. FO TRUTH TABLE

Step	Result
0	The coil is energized if the pointer equals 0, or is greater than the Table Length. In PC-1100, pointer is zeroed if greater than stack length.
↑	Data transfers from the table end position in the destination. The contents of the table shift down one place. The pointer decrements by 1. <sup>1</sup> In PC-1100, when table is shifted, contents of highest register in stack is duplicated leaving a trail of data. In the PC-1200, after table is shifted, highest register shifted is zeroed and erases data trail.
1	The coil energizes when the pointer equals zero.
<sup>1</sup> If the pointer value is zero, then zero is transferred to the Destination Register.	

**APPLICATIONS**

The FIFO Stack is used in a situation where a machine performs a controlled sequence with variable data. For example, a machine that paints an object that comes in several colors can be programmed for the day's production. The operator need only enter the color choices, and the program retrieves the choice for the machine in the proper order. (See Figure 4.)

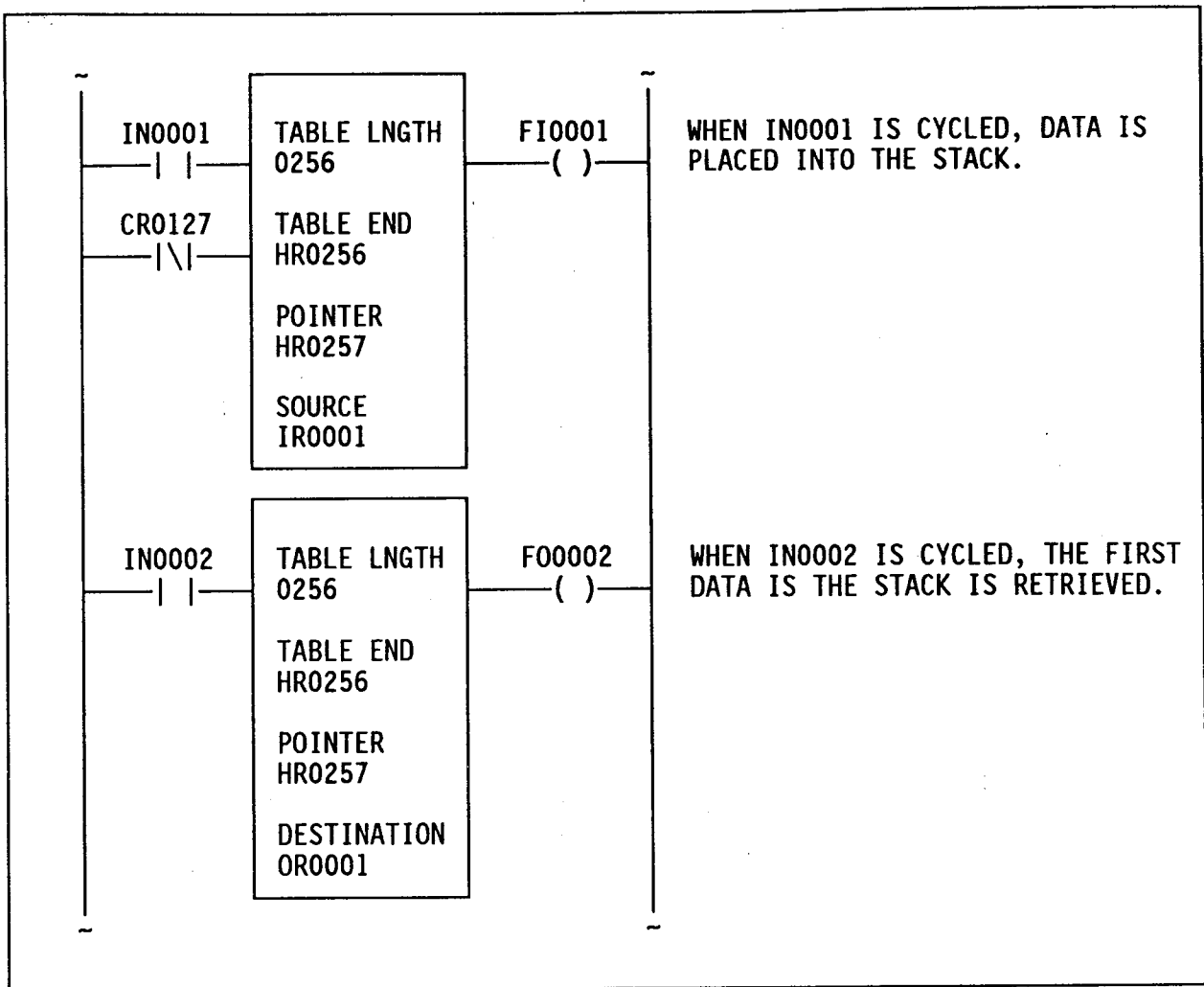


Figure 4. FIFO Stack Application

# FI/LO-FILO STACK

Modified for PC-1200

PC-1100-x01y: NOT SUPPORTED	PC-1100-x05y: SUPPORTED
PC-1100-x02y: SUPPORTED	PC-1200-x02y: SUPPORTED
PC-1100-x03y: SUPPORTED	PC-1200-x04y: SUPPORTED

## DESCRIPTION

The FILO Stack function is a combination of two functions: the First in Stack (FI) function, and the Last Out Fetch (LO) function. In the FI/LO functions, the first data entered is the last data retrieved. FI function symbology is shown in Figure 1; LO function symbology is shown in Figure 2.

### Note

FI and LO functions are designed to be used together.

Figure 3 uses a card playing analogy to demonstrate the FILO Stack concept. Individual cards are stacked on a table and dealt from the top of the deck. When Card No. 52 is removed, Card No. 51 occupies the top position. Card No. 1 is the last card removed.

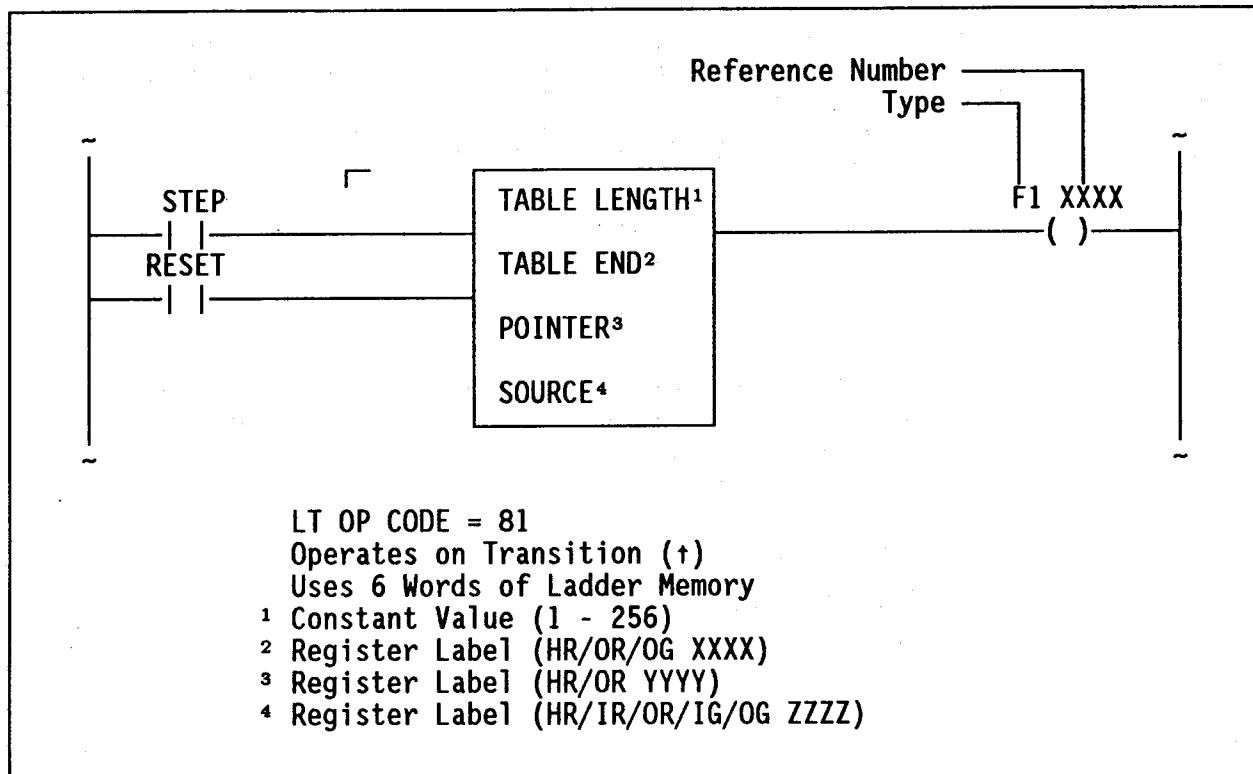


Figure 1. First in Stack (FI)

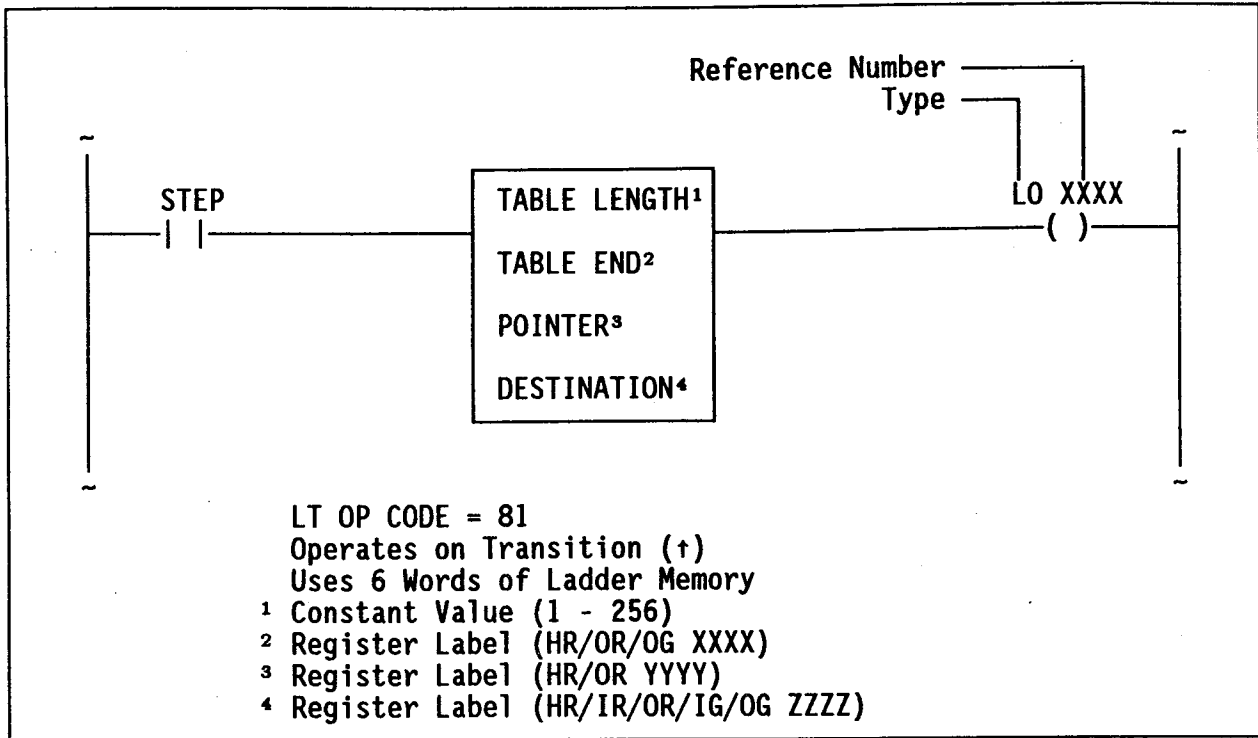


Figure 2. Last Out Fetch (LO)

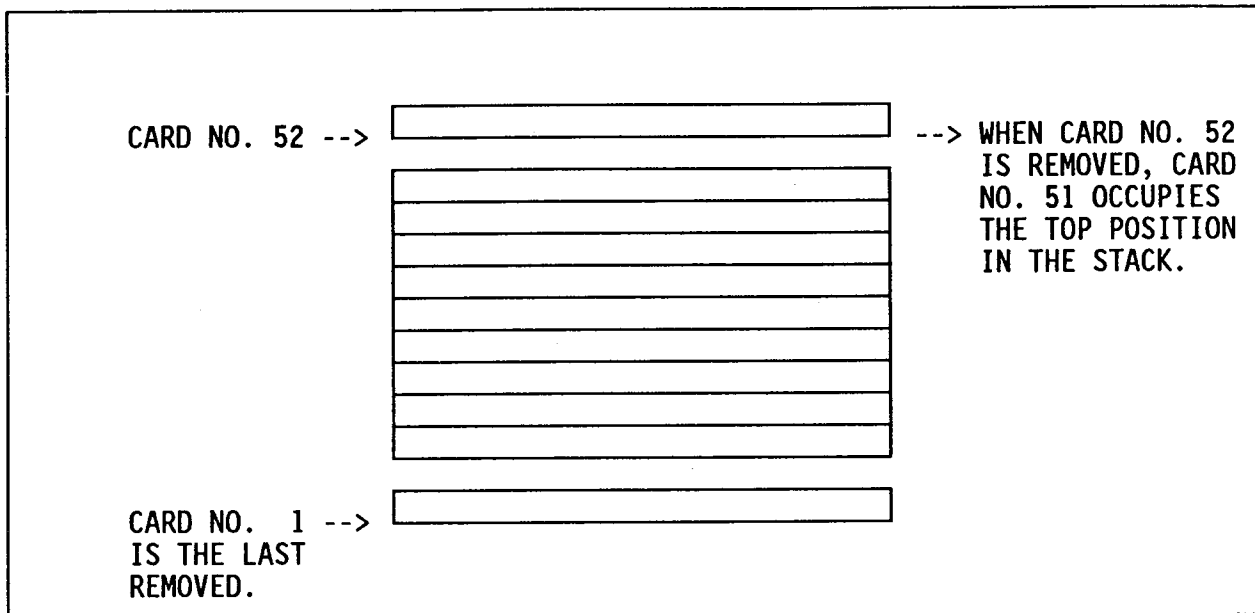


Figure 3. FILO Stack Concept

# FI/LO

## OP CODE

Op Codes 81 and 86 defines the Literal (LT) as either a FI or LO function. Whether or not the Literal function should be used depends upon the capability of your program loader. It is recommended that the mnemonic function be used whenever possible. Refer to the Introduction in this Section and the LT function description.

## SPECIFICATIONS

### OPERAND 1 - TABLE LENGTH

The table length is a constant value that defines the size (i.e., number of registers) of the FILO Stack. Both functions should be programmed with the same table length. The size of the FILO Stack is limited, as specified in Table 1, with a maximum of 256.

TABLE 1. TABLE LENGTH AND TABLE END LIMITS

Type	PC-1100	PC-1200-1020/1040	PC-1200-1041/1041/1042	PC-1250
HR	$\leq 1792$ <sup>1</sup>	$\leq 1792$	$\leq 1792$	$\leq 1792$
OR	$\leq 8$	$\leq 32$	$\leq 64$	$\leq 128$
OG	$\leq 8$	$\leq 32$	$\leq 64$	$\leq 128$

<sup>1</sup> For the PC-1100, the maximum number of holding registers depends on memory size, as described in Section 4.

### Note

The highest Holding Register reference number acceptable is dependent on the memory and user program size.

### OPERAND 2 - TABLE END

The table end defines the type and number of the last register or group in the FILO Stack and is subject to the limits in Table 1. Both functions should be programmed with the same program end.

### OPERAND 3 - POINTER

The pointer is a register that contains the number of items currently in the FILO stack. This register must be the same in both functions:

- Holding Register (HR)
- Output Register (OR)

**OPERAND 4 (FI) - SOURCE**

In the FI function, the source defines the location from which data is moved to the stack. This location is a specified register or group:

- Holding Register (HR)
- Input Register (IR)
- Output Register (OR)
- Input Group (IG)
- Output Group (OG)

**OPERAND 4 (FO) - DESTINATION**

The destination is used in the LO function to define the location to which data is moved from the FILO Stack. This location is a specified register or group:

- Holding Register (HR)
- Output Register (OR)
- Output Group (OG)

**FI TRUTH TABLE**

See Table 2.

**TABLE 2. FI TRUTH TABLE**

Step	Reset	Result
X	0	The coil de-energizes. The pointer is set to zero.
↑	1	The source moves to the position in the stack designated by the pointer. After the move, the pointer increments by 1. See description below.
X	1	The coil energizes when the pointer value equals the stack size and de-energizes when the pointer value is less than the stack size.
X = Don't Care		

## FI/LO

The stack grows from the highest numbered holding register (stack end) toward the lower numbered registers. The pointer always points to the next available stack location. A pointer value of zero points to the first stack location (stack end). A pointer value of N-1, where N is the specified stack length, points to the last available stack location. A pointer value equal to the stack length implies a full stack.

## LO TRUTH TABLE

See Table 3.

TABLE 3. LO TRUTH TABLE

Step	Result
0	The coil is energized if the pointer equals 0 or is greater than the Table Length. In the PC-1100, the pointer is zeroed if greater than table length.
↑	The pointer decrements by 1. The destination transfers data at the pointed location. <sup>1</sup>
X	The coil energizes when the pointer equals zero.
<sup>1</sup> If the pointer value is zero, then zero is transferred to the Destination Register.	
X = Don't Care	

## APPLICATIONS

The FILO Stack function monitors a system(s) for fault conditions and status when a fault occurs. Assuming that the system in question resets itself in the case of non-critical faults, the monitoring processor maintains a fault/fault status history. When hard or recurring faults occur, machine status is recalled in reverse order, the most recent fault status is recalled first. The processor is not part of the monitored system and is unaffected by system failures.

As shown in Figure 4, the status of selected switches and contacts is entered into the processor via IR0001 when a fault occurs in the operating system.

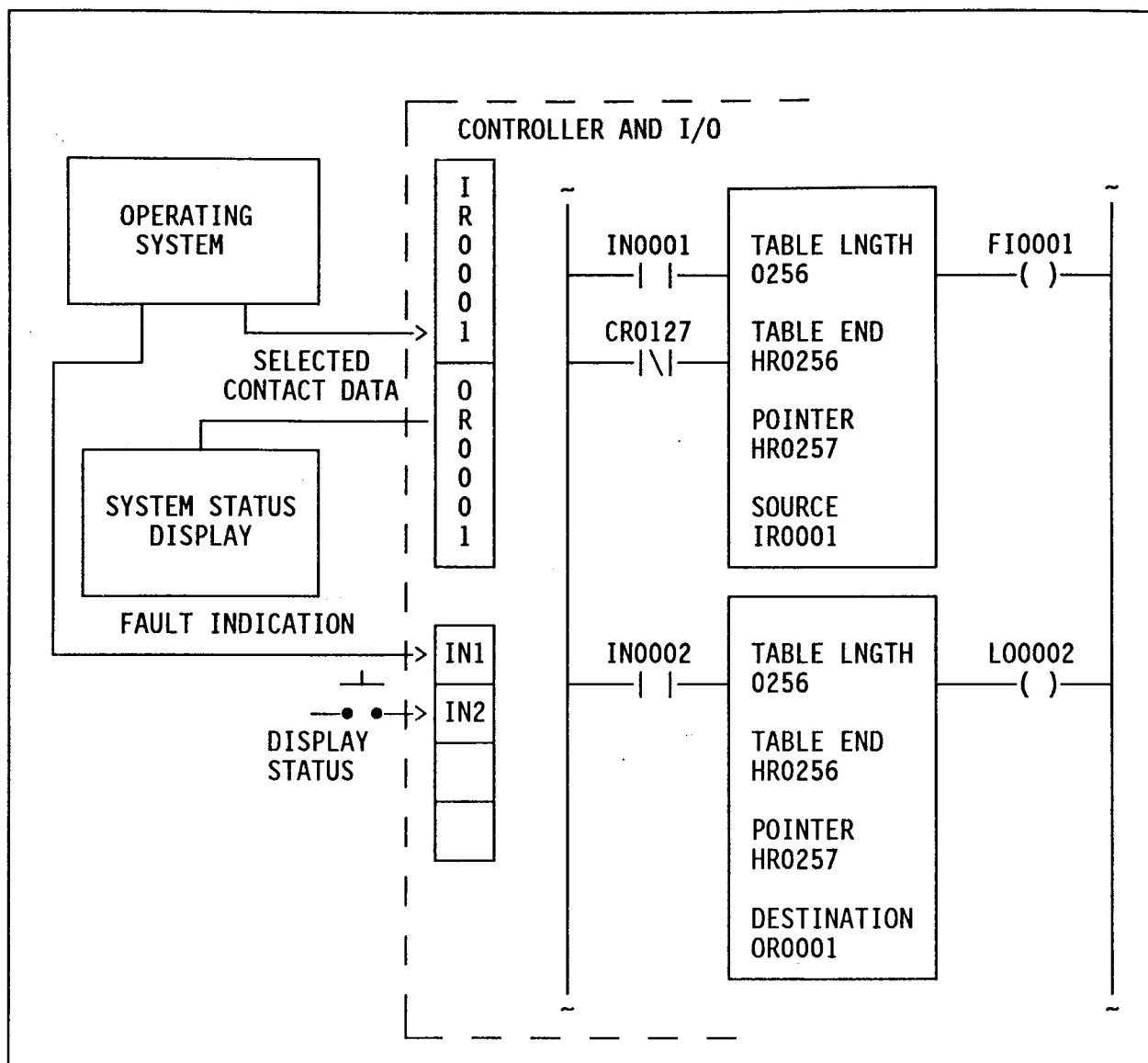


Figure 4. FILO Stack Application



# IM - INDIRECT MOVE

PC-1100-x01y: SUPPORTED  
 PC-1100-x02y: SUPPORTED  
 PC-1100-x03y: SUPPORTED

PC-1100-x05y: SUPPORTED  
 PC-1200-x02y: SUPPORTED  
 PC-1200-x04y: SUPPORTED

## DESCRIPTION

The Indirect Move (IM) function allows the contents of the source register to be duplicated in locations in a pre-defined table of registers with the relative location being determined by the contents of a pointer location. Figure 1 illustrates the concepts and symbols related to the Indirect Move function. The operation of the function is depicted in a simplified fashion in Table 1. When studying the Table, assume the following:

- Table Length = 10
- Table End = HR0010
- Pointer = HR0011
- Source = IR0001

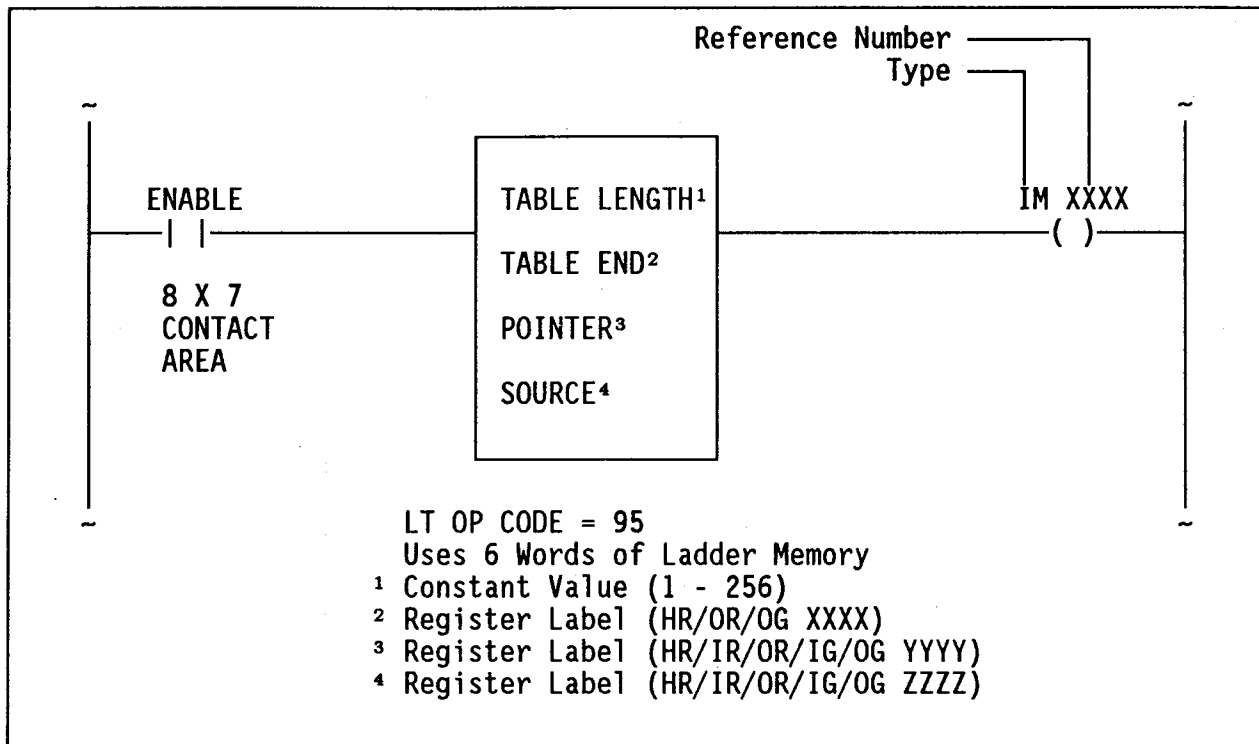


Figure 1. Indirect Move Function Concept

Only an enable circuit is used with this function. With the Indirect Move enable circuit conducting, data is transferred from the source register to the register specified by the pointer. The Indirect Move coil is energized when an invalid pointer value is received. If the coil is forced, only its contacts (and output circuit, if any) are affected; the Indirect Move function continues to operate according to the enable circuit.

TABLE 1. IM OPERATION

Source (IR0001)	Pointer (HR0011)	Holding Register Contents	
1000	00000	HR0001	1000
2200	00001	HR0002	2200
0016	00002	HR0003	0016
1711	00003	HR0004	1711
0296	00004	HR0005	0296
0011	00005	HR0006	0011
7942	00006	HR0007	7942
0573	00007	HR0008	0573
7184	00008	HR0009	7184
9999	00009	HR0010	9999

## OP CODE

Op Code 95 defines the Literal (LT) as being an Indirect Move (IM) function. Whether or not the Literal function should be used depends upon the capability of your program loader. It is recommended that the mnemonic function be used whenever possible. Refer to the Introduction in this Section and the LT function description.

## SPECIFICATIONS

### OPERAND 1 - ENABLE CIRCUIT

When conducting data is copied from the source register to the destination register (specified by the pointer) during each processor scan.

### OPERAND 2 - TABLE LENGTH

A constant value defining the number of registers in the table. The table may be from 1 to 256 registers in length.

### OPERAND 3 - TABLE END

Defines the type and number of the last register in the destination table, in accordance with Table 2.

TABLE 2. LAST REGISTER TYPE, NUMBER

Type	PC-1100	PC-1200-1020/1040	PC-1200-1041/1041/1042	PC-1250
HR	$\leq 1792$ <sup>1</sup>	$\leq 1792$	$\leq 1792$	$\leq 1792$
OR	$\leq 8$	$\leq 32$	$\leq 64$	$\leq 128$
OG	$\leq 8$	$\leq 32$	$\leq 64$	$\leq 128$

<sup>1</sup> For the PC-1100, the maximum number of holding registers depends on memory size, as described in Section 4.

**OPERAND 3 - POINTER**

Contains the current table location into which data is transferred. Pointer contents of zero references are first (lowest register) and contents of table length minus 1 selects the table end register. This may be a specified:

- Holding Register (HR)
- Input Register (IR)
- Output Register (OR)
- Input Group (IG)
- Output Group (OG)

**OPERAND 4 - SOURCE**

The location from which data is transferred. This may be a specified:

- Holding Register (HR)
- Input Register (IR)
- Output Register (OR)
- Input Group (IG)
- Output Group (OG)

**COIL**

Coil is energized when the enable circuit is conducting and the pointer value is greater than the Table Length minus 1. No data is transferred when the pointer value is greater than the Table Length minus 1. The function still operates if the coil is forced on.

## APPLICATIONS

The Indirect Move (IM) function is useful when process variables in the system under control are changed at various times to handle new system demands. For example, assume that the system under control is a heat-treating process. The presets to a number of timers are changed each time a new item is run through it. Using the system shown in Figure 2 and the program in Figure 3, the changing of the preset times is easily accomplished.

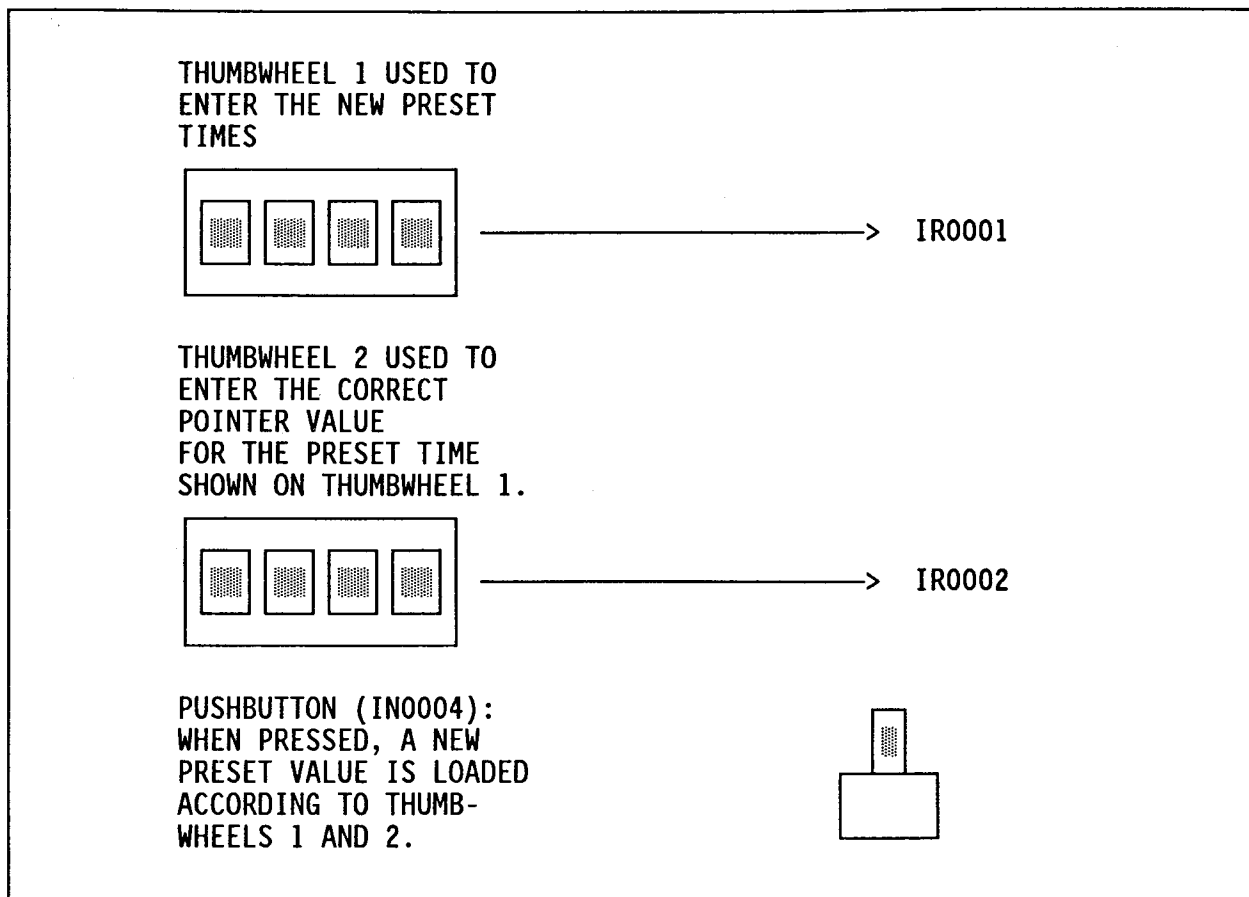


Figure 2. Process Input Data

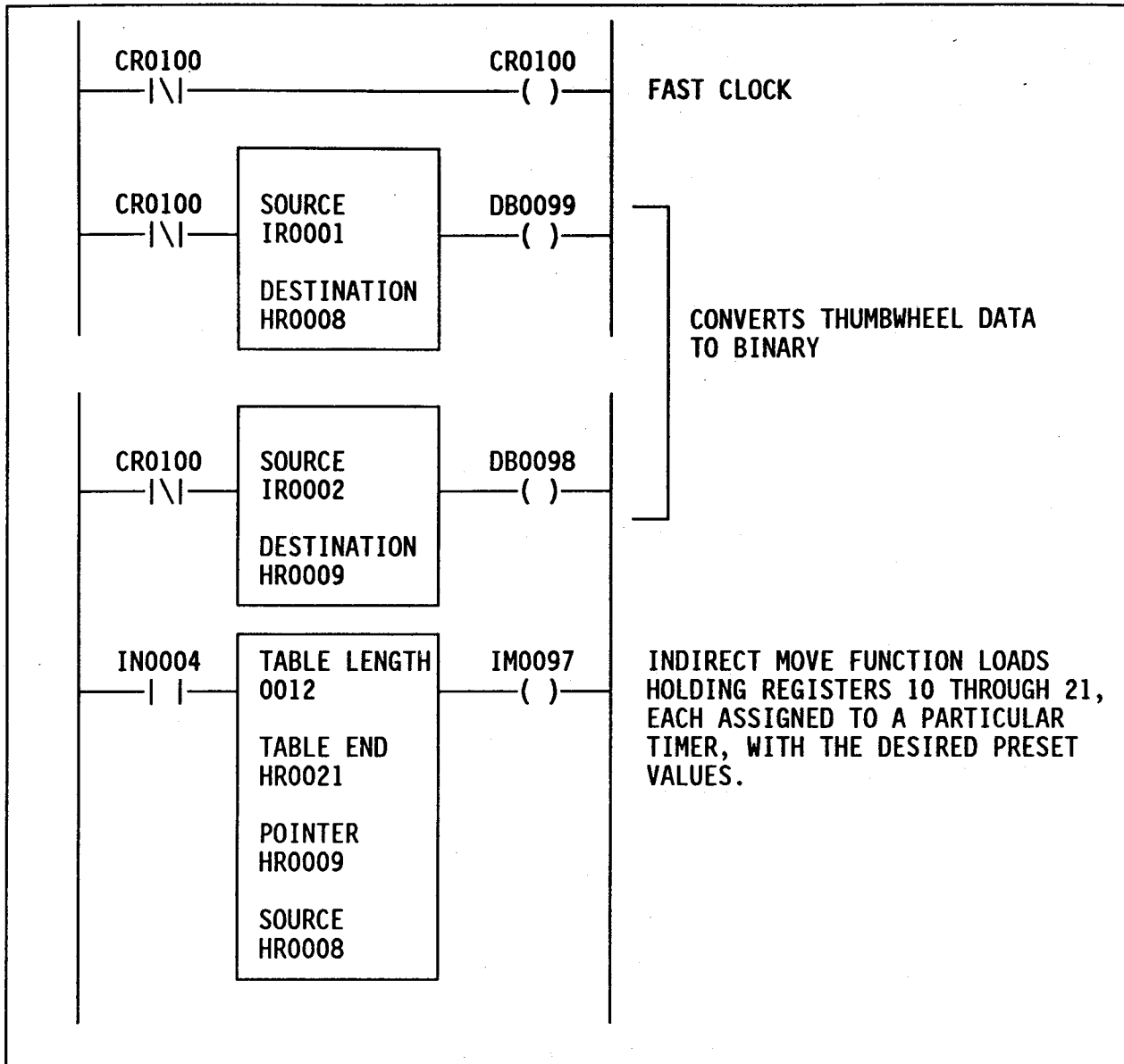


Figure 3. Program Changing Preset Times

# LC - LOOP CONTROLLER

Modified for PC-1200

PG-1100-x01y: NOT SUPPORTED	PG-1100-x05y: NOT SUPPORTED
PG-1100-x02y: NOT SUPPORTED	PG-1200-x02y: NOT SUPPORTED
PG-1100-x03y: SUPPORTED	PG-1200-x04y: SUPPORTED

## LOOP CONTROL THEORY AND APPLICATION

This introductory section describes the theory and operational concepts of loop control. If you are already familiar with loop control theory, the Loop Control (LC) special function description begins on page 5-128.

The Loop Controller (LC) function implements user-defined, closed-loop process control systems. Closed-loop systems, in general, are characterized by an ability to compare the actual value of some process variable with its desired value and to take the necessary corrective action.

Regardless of the method (type of controller) used to control the process, all closed-loop systems share the same basic block diagram. (See Figure 1).

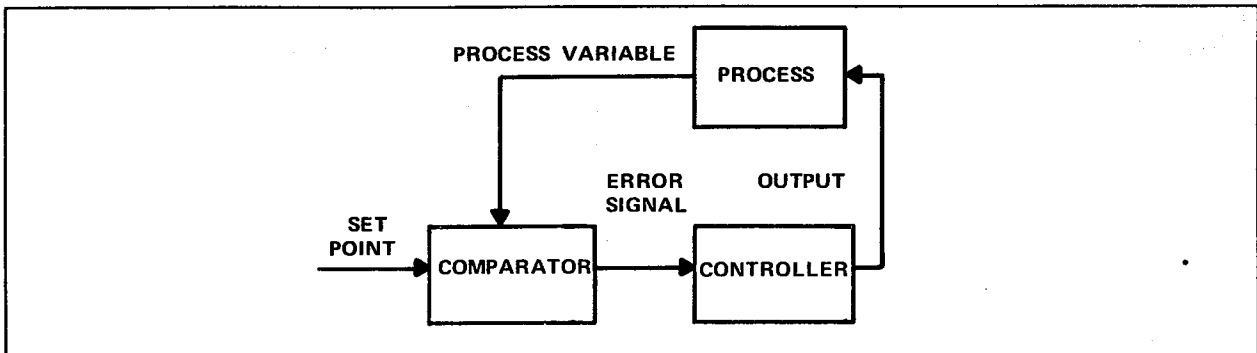


Figure 1. Closed-Loop System

Consider the manual closed-loop system shown in Figure 2.

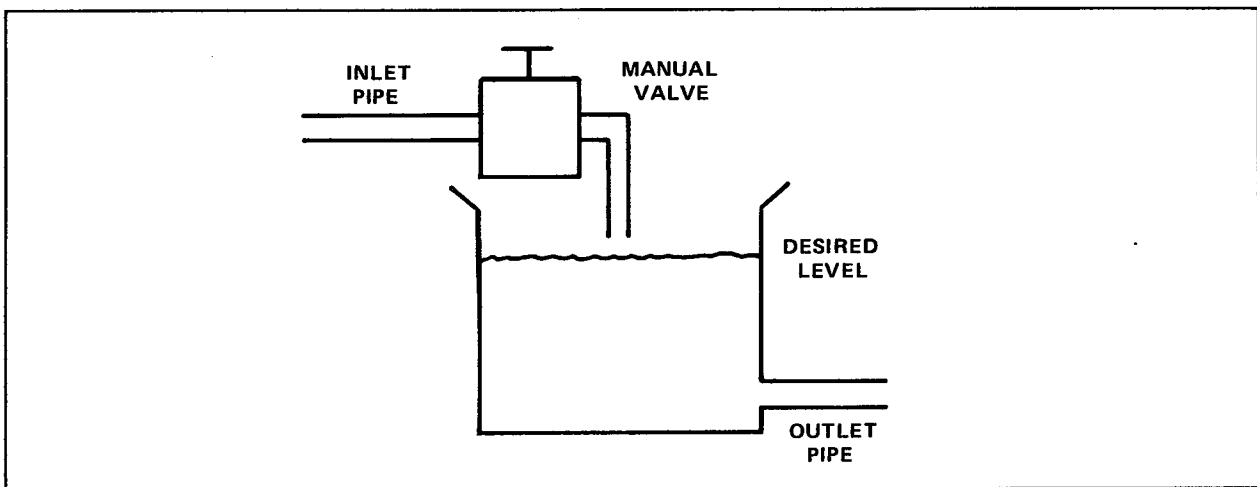


Figure 2. Manual Closed-Loop System

As shown, this system operates in an open-loop fashion (i.e., the process variable, liquid level, is not actively compared to the desired level, and no spontaneous corrective action is taken to maintain the desired level). This system could become closed-loop in operation by adding an operator. The operator closes the loop by fulfilling the roles of the comparator and the controller. The operator sees that the actual height is not the desired height, and adjusts the flow from the inlet pipe to keep the liquid level at its desired height. However, this is a very inefficient method of control in a rapidly changing environment. This same system could be automated, as shown in Figure 3.

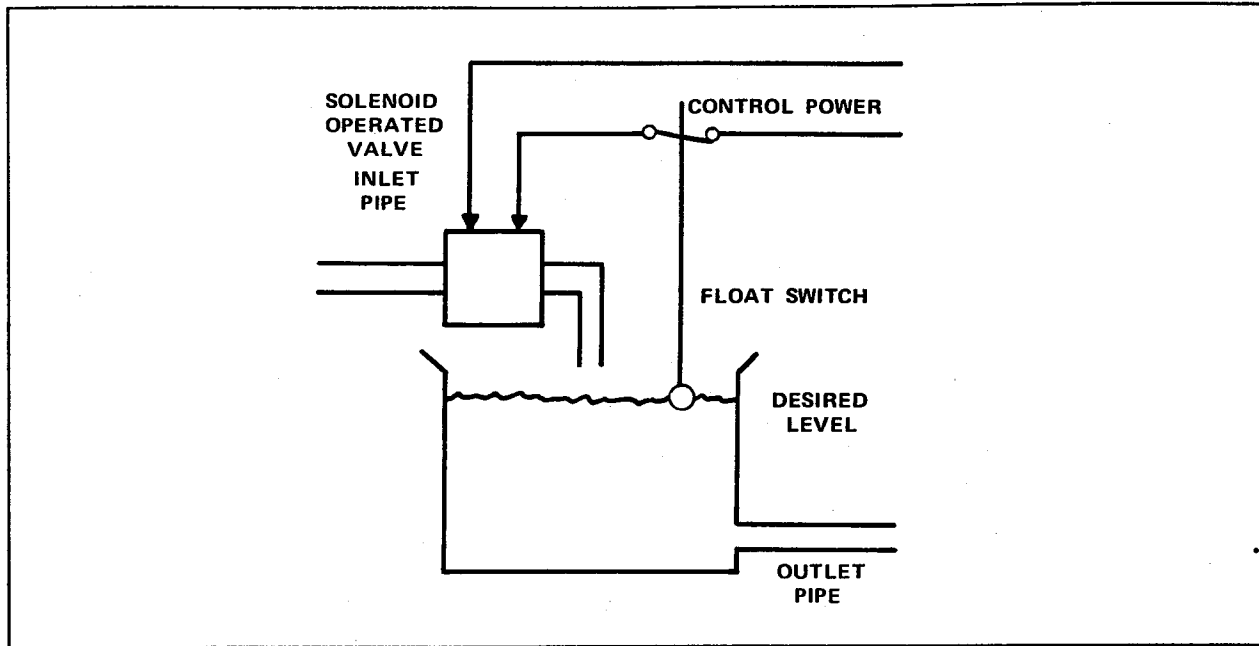


Figure 3. Simple Closed-Loop System

As shown, the simple closed-loop system operates to keep the liquid level at the desired point by turning the inlet valve ON whenever the float switch closes, and turning the valve OFF when the float switch opens. This type of system is efficient only as long as the process variable, liquid level, does not vary rapidly and/or the desired height of the liquid level does not change. Rapid variations in the level will cause frequent operation of the solenoid valve and float switch, leading to early failure and frequent replacement of these components. The desired height of the liquid level can only be changed by modifying the physical position of the float switch (possibly a time-consuming and expensive operation).

A better system is shown in Figure 4. It is more sophisticated, but it is also more efficient.

As shown in Figure 4, the better closed-loop system operates to maintain the desired level by adjusting the input to correct for any variations without a great deal of oscillation. The set point can be easily modified by changing a setting, rather than readjusting the float switch. The valve responds to the appropriate error size (e.g., a small error results in a small flow change; a large error results in a large flow change). The actual versus desired level is closely tracked by the system.

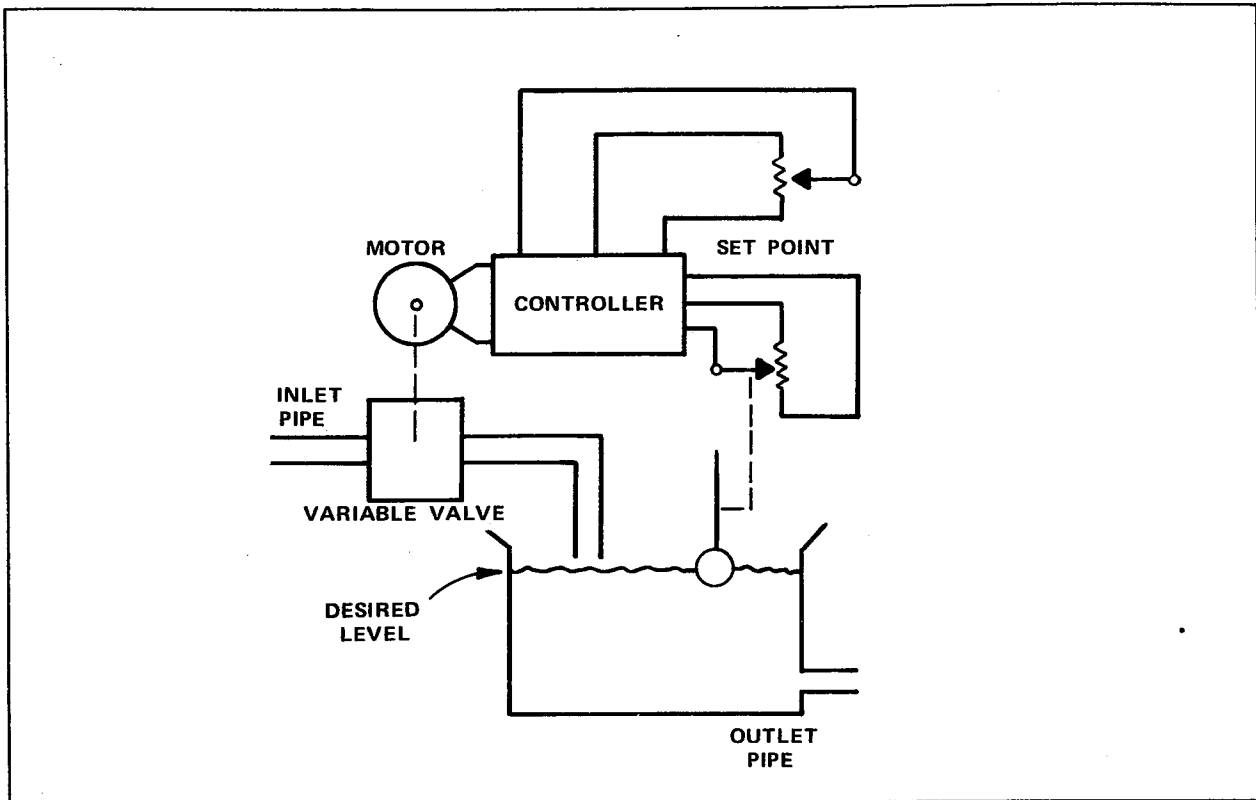
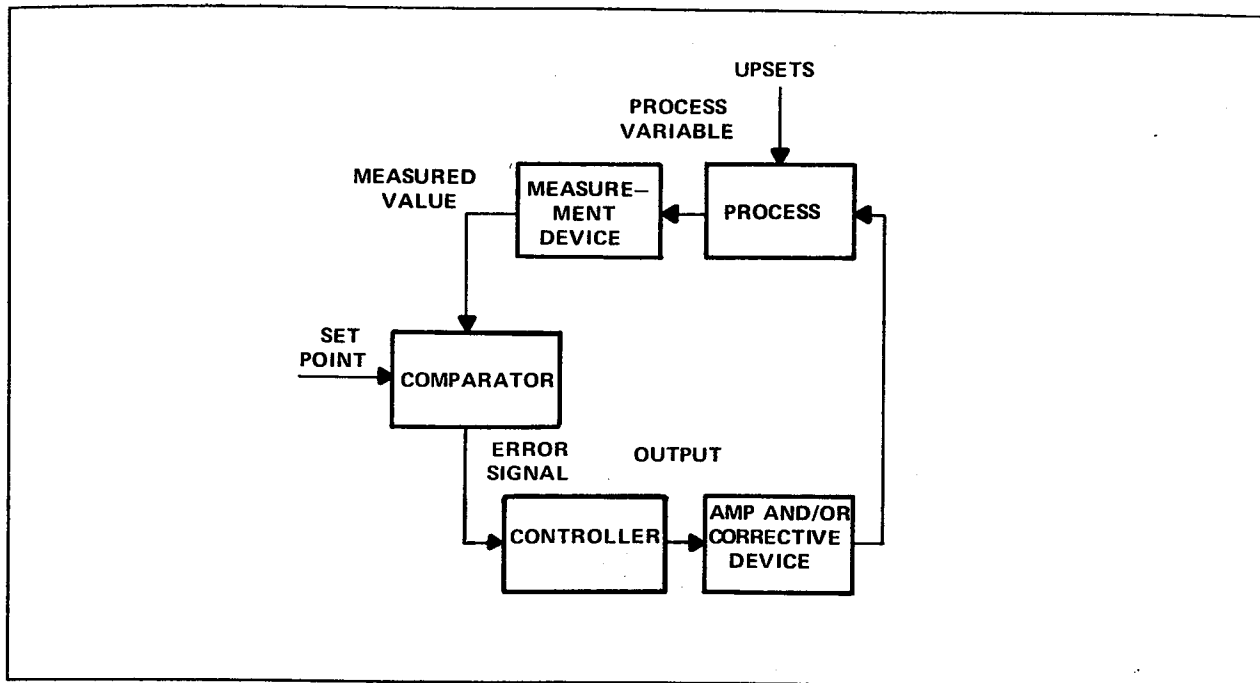


Figure 4. Better Closed-Loop System

The closed-loop system general block diagram of Figure 1 can be made more specific, as shown in Figure 5.

Generally, closed-loop systems operate as follows: when the system is operating in a stable fashion, the set point minus measured value results in a zero error signal and a zero response from the controller; this further results in no changes to the process and no change in the process variable or measured value. In essence, the system is operating as desired. When the process changes (e.g., increased flow requirement from the outlet pipe), the process variable changes, causing the measured value to change. This difference between set point and the measured value causes an error signal to be sent to the controller. The controller output moves a correcting device to correct for the disturbance. This causes the process variable to return to a level where it equals the set point, thereby, cancelling the error signal to restore stable operation.





**Figure 5. Detailed Closed-Loop System Block Diagram**

There are several system characteristics that define a good closed-loop system (i.e., how well the system reduces the error signal to zero or to almost zero). The final difference between the measured value and set point (stable operation) is called "offset".

- A good system has a low offset.

A second, and possibly, more important characteristic is the speed with which a system responds or restores agreement.

- A good system has a fast speed of response.

Further, the system should be free of large and violent oscillation.

- A good system is stable.

With these characteristics in mind, it should be apparent that all systems cannot meet all requirements with a single type of controller. There are five recognized modes of control.

- ON-OFF control  
(Simple systems - LC function not required)
- Proportional control  
(Magnitude-oriented control)

(Magnitude-oriented control)

- Proportional-plus-Integral (PI) control

(Magnitude and error time duration oriented)

- Proportional-plus-Derivative (PD) control

(Magnitude and error rate of change oriented)

- Proportional-plus-Integral-plus-Derivative (PID) control

(Magnitude, error time duration, and error rate of change oriented)

These control modes increase in complexity from the top to the bottom of the list. Usually, the more difficult the control problem, the farther down the list the solution is found. Since ON-OFF control systems are generally not handled by the LC function, this control mode is not discussed in this document.

### PROPORTIONAL CONTROL ( $M = K_c e$ )

In the Proportional control mode, the final correcting device is operated over a continuous range of possible positions. The exact position is proportional to the error signal (i.e., the output of the controller is proportional to the error signal).

In the Proportional control system shown in Figure 6, the final correcting device is a variable position valve controlling the fuel supply to a burner that is used to heat a product passed through a combustion chamber. The system operates as follows: when the valve opens, the temperature increases; when the valve closes, the temperature decreases. Assume that the controller is set up such that when the temperature of the product is 190°F or higher, the controller will fully close the fuel valve, and when the temperature is 165°F or lower, the controller will fully open the fuel valve. The system, however, may be assumed to be operating at a steady-state temperature of 180°F and the valve 40 percent open. (Due to the many unpredictable factors in maintaining operation at 180°F, actual valve position cannot be predicted; therefore, 40 percent is an assumption.) Figure 7 shows the conditions defined for this system.

Assume that a change causes the measured temperature to drop to 175°F. The valve will open to 60 percent of its range, causing temperature to rise accordingly. If the change had been larger (to 170°F), the valve would have opened to 80 percent of its range. This change illustrates the proportional nature of the system. A 5°F drop in temperature results in a 20 percent increase in valve opening; a 10°F drop in temperature results in a 40 percent increase in valve opening (i.e., doubling the error doubles the response, a proportional reaction.)

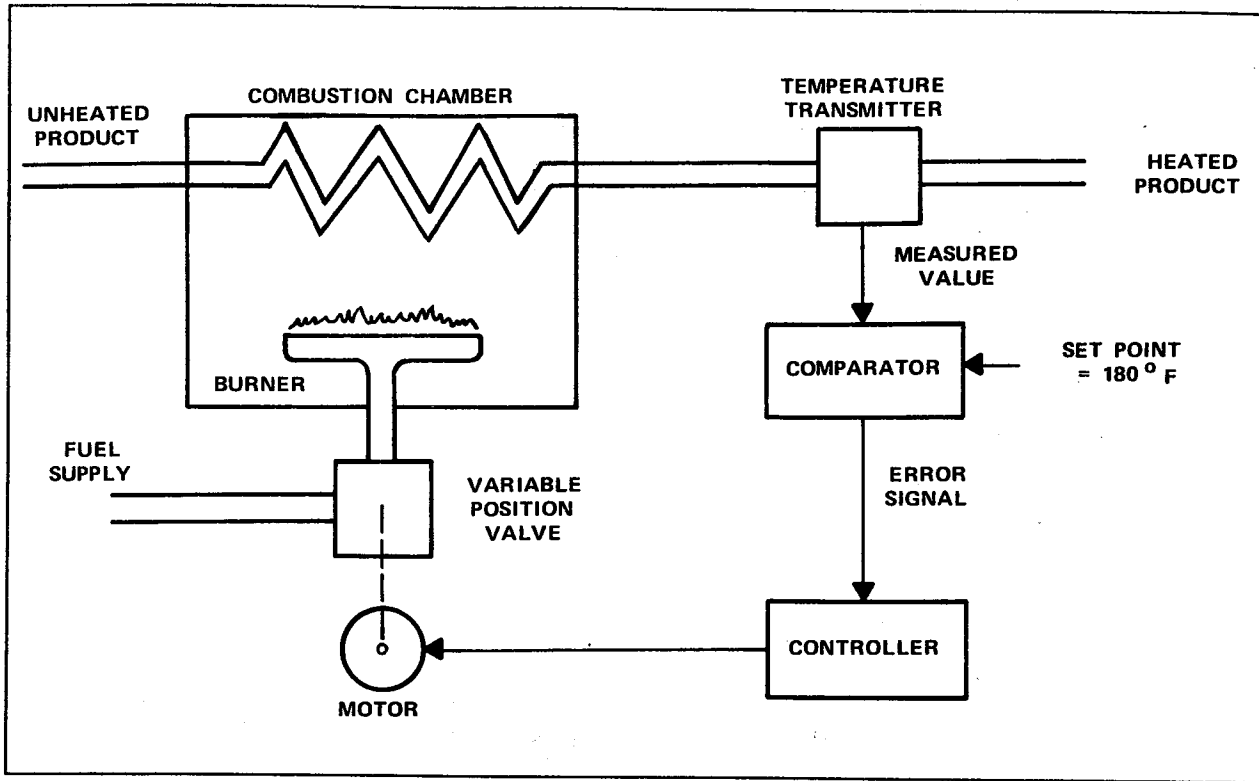


Figure 6. Proportional Control System

**PROPORTIONAL BAND**

Figure 7 shows that the valve is fully-open at 165°F and fully-closed at 190°F (a range of 25°F). This range is called the proportional band of control.

Within this band of temperature, valve response is proportional to temperature change. Proportional band is expressed as a percentage of full scale range of the controller. If, for example, the set point can be between 60°F and 300°F, the proportional band is:

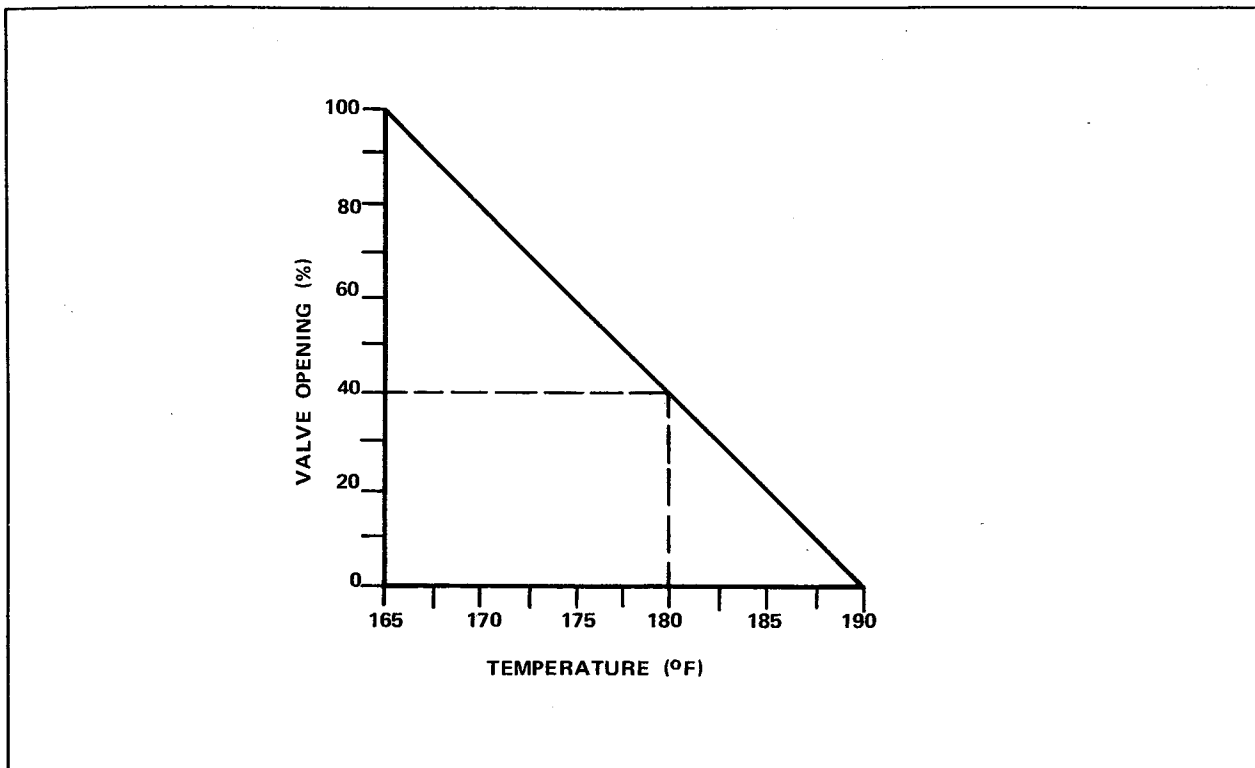
$$\frac{25^\circ\text{F}}{(300^\circ\text{F} - 60^\circ\text{F})} = 0.104 = 10.4 \text{ percent}$$

Formally defined, proportional band is the percentage of full controller range by which the measured value must change to cause the correcting device to change by 100 percent. Since the proportional band is easily defined, we can derive the gain by using the following calculation:

$$\text{Gain } (K_c) = \frac{100}{P}$$

where:

P = proportional band, expressed as a percentage



**Figure 7. Operating Range/Set Point**

Using the information given in Figure 7:

$$\text{Gain } (K_c) = \frac{100}{10.4} = 9.62$$

The system shown in Figure 7 can then be said to have a proportional band of 10.4 percent and a proportional gain of 9.62. Given the same proportional band and the same operating condition of 180°F, conditions may be such that a 60 percent valve opening is required to maintain 180°F. The slope of the graph would be the same as in Figure 7, but the upper and lower limits will have changed (i.e., increased by 5°F). (See Fig. 8.)

Control will remain proportional, but the valve will now be fully-closed at a measured temperature of 195°F, and fully-open at a measured temperature of 170°F. Everything has shifted upward accordingly.

Proportional control tends to eliminate permanent system oscillations found in more simple ON-OFF control schemes (such as that shown in Figure 3). Oscillations are still possible, but they should tend to die out. It is also possible to set the proportional band narrow enough to cause the system to act (and oscillate) as if it were an ON-OFF system. A properly-designed Proportional control system is still relatively simple, but overcomes the inherent problems of oscillation and part wear found in ON-OFF systems.

Response in a Proportional control system is a function of proportional band. Consider the three response graphs shown in Figure 9.

In the system shown in Figure 6, different proportional band settings result in different responses.

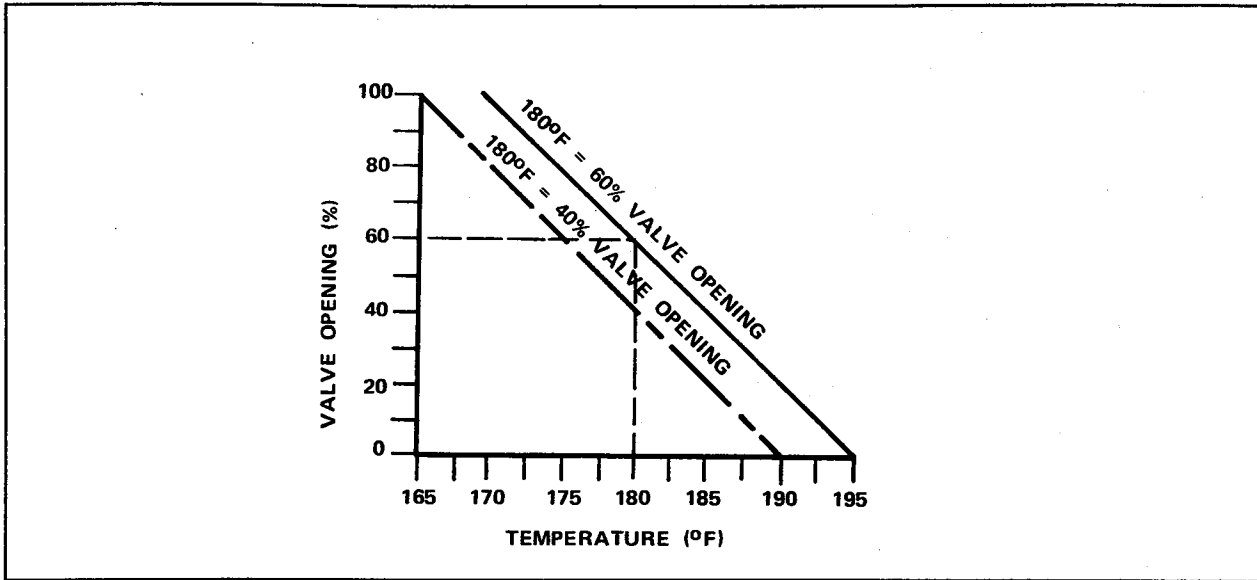


Figure 8. Operating Range/Set Point with Redefined Limits

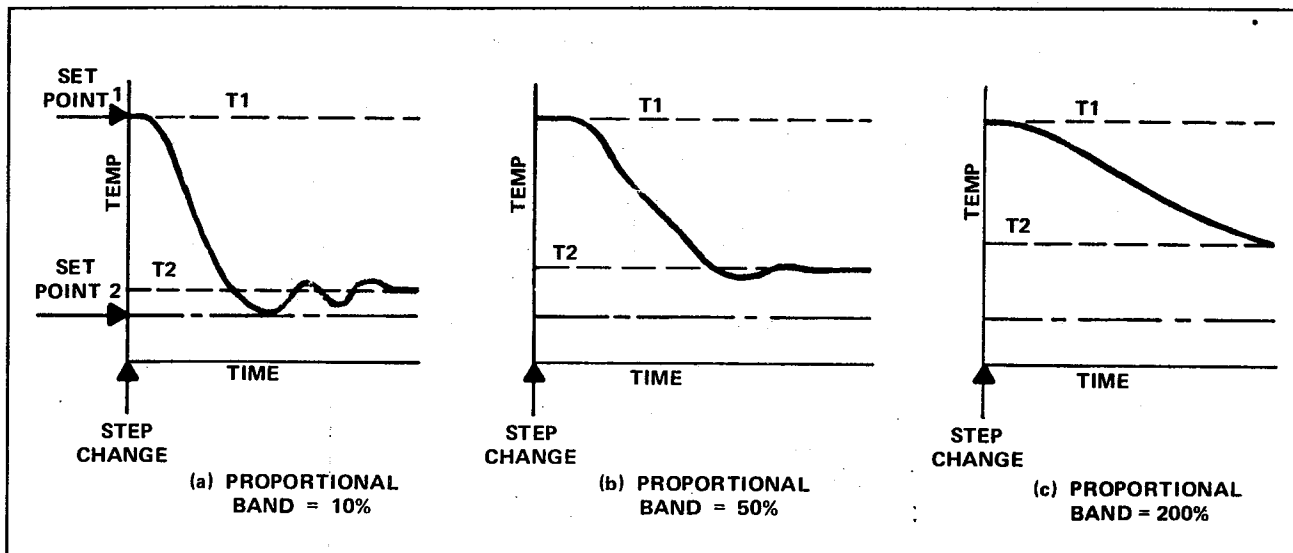


Figure 9. Effect of Proportional Band

Figure 9 shows system reaction to a set-point change, causing the temperature to change from  $T_1$  to  $T_2$ . In (a), a relatively narrow proportional band setting of 10 percent shows that the system responds rapidly to changes, but oscillates once at the desired position. In (b), at 50 percent proportional band, the response is slower, with less oscillation. Finally, in (c), the response is much slower with no oscillations; control is smoother.

### STEADY-STATE ERROR (OFFSET)

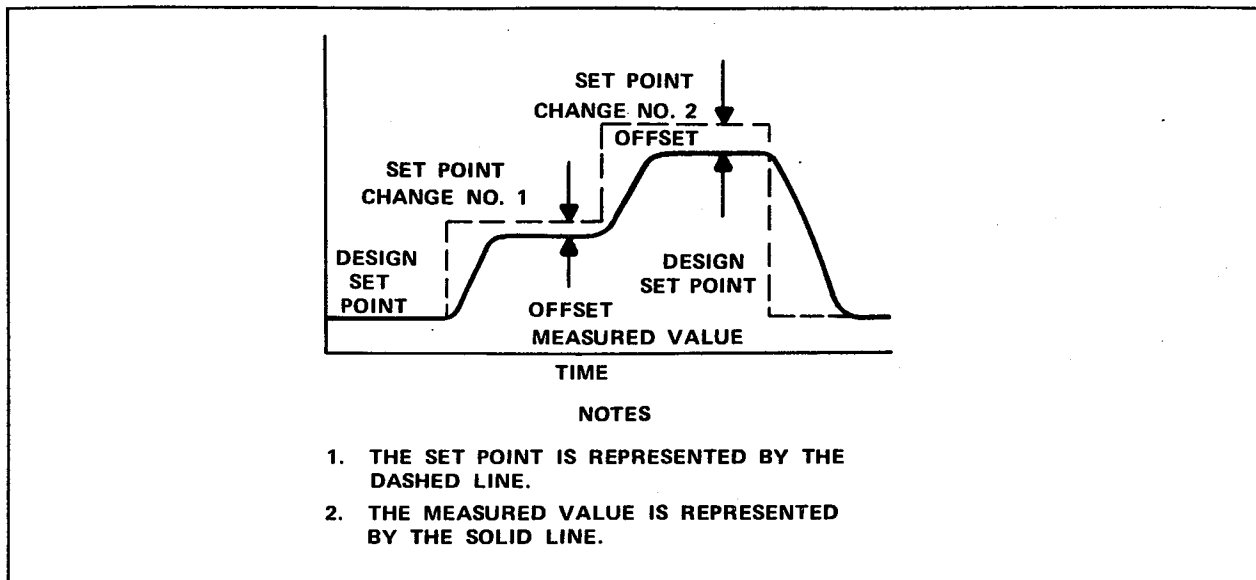
Refer again to the graphs of Figure 9. Note that in the plot of temperature versus time, the actual measured temperature does not reach the desired control value. The wider (larger) the proportional band, the greater the difference between the final temperature ( $T_2$ ) and the new set point (Set Point 2).

To understand this effect, re-examine the operation of the proportional system used in Figure 6. When the controller is maintaining  $180^\circ\text{F}$  with a valve opening of 40 percent and a disturbance of some nature occurs causing the temperature to drop, the valve will drive open to increase the fuel flow to correct the drop in temperature. The control valve must now remain further open permanently to maintain heat at the desired level. Since the percentage of valve opening is proportional to the error signal, a permanently increased valve opening can only occur if there is a permanently increased error. Proportional systems tend to have a permanent error. The more narrow the proportional band, the smaller the error. Proportional systems work very well where changes are small and slow, allowing a very narrow proportional band, and a very small permanent error.

### BIAS ( $M = M_o + K_c e$ )

In purely proportional control,  $M = K_c e$ . If  $e$  equals zero,  $M$  (the output) must also be zero. To achieve a zero error with a non-zero output, we must introduce a bias term,  $M_o$ , such the  $M$  equals  $M_o$  at  $e$  equals zero. Assume that a zero error signal occurs when the valve is 40 percent open and the temperature is  $180^\circ\text{F}$ . If the system was started, the error signal is a large negative value and the valve would be open more than 40 percent. This implies a 40 percent bias. As the temperature more closely approaches  $180^\circ\text{F}$ , the more closely the valve approaches 40 percent open, and the system becomes stable at  $180^\circ\text{F}$ . In this system, this is the only possible set point at which there will be a zero error. To have a zero error, a unique set point. If  $M_o$  is constant at 40 percent, and a new set point is selected (e.g.,  $185^\circ\text{F}$ ), the valve would have to open more than 40 percent and the error signal can no longer be zero. Actual measured temperature could not quite reach  $185^\circ\text{F}$  to maintain the error necessary to keep the valve open more than 40 percent. This difference is the offset and the farther the set point is from the "zero-error" set point, the worse the offset.

Consider the graph shown in Figure 10.



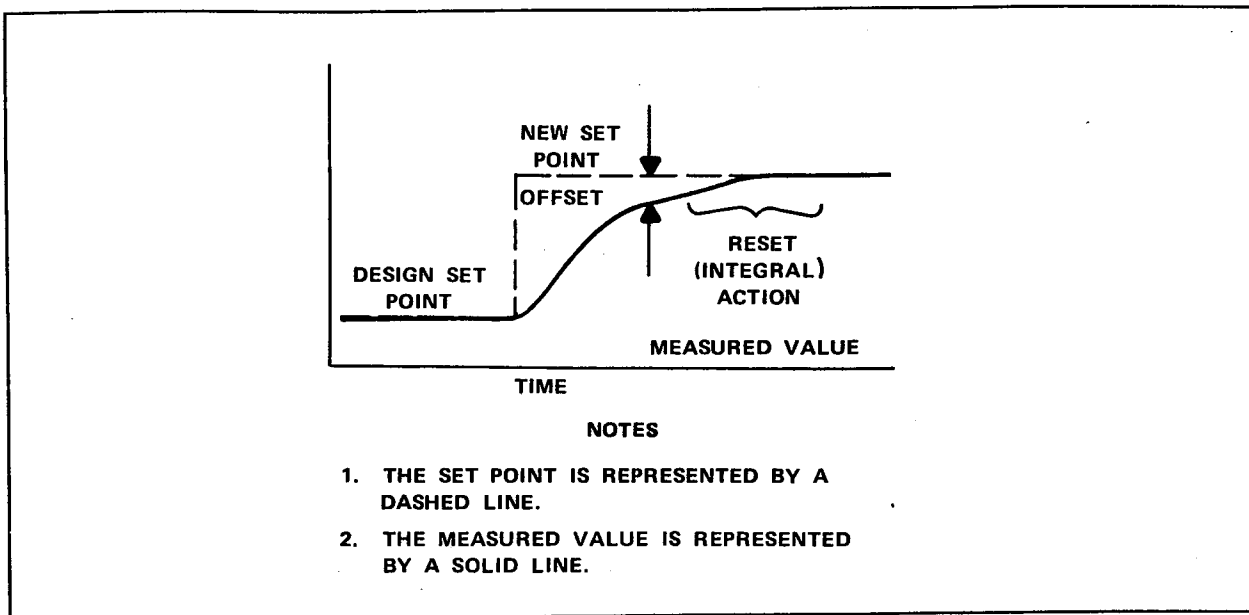
**Figure 10. Offset Effect**

As shown, the proportional system operates without offset when it is operating at the exact design set point. If the set point is changed for any reason, offset occurs, increasing in size as the difference between the design set point and new set point increases. Offset results in a constant error signal (set point minus measured value) for all conditions other than the design set point. Strict proportional control can be best used where load changes are small and low, and set point variation is small.

### PROPORTIONAL-PLUS-INTEGRAL (PI) CONTROL ( $M = K_c e + K_i I$ )

Systems subject to large changes do not function well in a purely Proportional mode due to this offset problem. The method of overcoming the offset problem is to have an automatic function that senses the offset and continuously recalculates the bias such that the error can be zeroed (see Figure 11.)

To achieve this function, the time integral of the error signal (magnitude of the error multiplied by the time it has persisted) assists in determining the final valve (operating) level. This is Proportional-plus-Integral (PI) or Proportional-plus-Reset control. The proportional part positions the correcting device in proportion to the error that exists; then, the integral (or reset) part senses the offset that remains and continues motion of the correcting device in the same direction until the offset is reduced. In most applications, the integral time is not used directly; rather, the reciprocal of integral time is used and is termed the reset rate.



**Figure 11. Integral Control**

When the reset rate is low (large time constant), the integral part is slow to make its effects felt on the process.

The PI mode of control can handle most process control situations. Large loads and large variations in set point can be controlled efficiently with no prolonged oscillation, no permanent offset, and relatively quick recovery after an upset.

PI control can handle all control situations, except those having very rapid load changes and a long time lag between application of the corrective action and appearance of the corrective action in the process variable measurement.

### DERIVATIVE CONTROL

Since the derivative of any function is the measure of rate of change, this can be added to proportional control to cause the system to "overreact" to rapid (high rate of change) load variations. This is called derivative or rate control. These cases can be handled very effectively by the Proportional-plus-Integral-plus-Derivative (PID) control. In this mode, corrective action is determined as follows:

- Magnitude of the error (Proportional mode)
- Time duration of the error (Integral mode)
- Rate of change of the error signal (Derivative mode)



The effects of PID control are illustrated in Figure 12.

The PID control mode efficiently controls systems with wide variable set points, large load changes, and rapid system fluctuation.

Proportional-plus-Derivative (PD) systems are rarely used in process control systems; however, they are useful in position (servo) control applications. In these situations, the system responds to both the magnitude and the rate of change of the signal (in a system that can find a true null). (See Figure 13.)

In PD systems, an order signal is applied to the synchro control transformer, developing an error signal which is amplified, causing a motor to drive the signal to null by moving the rotor of the synchro control transformer. Using the PD mode allows faster response to large, rapidly changing position signals.

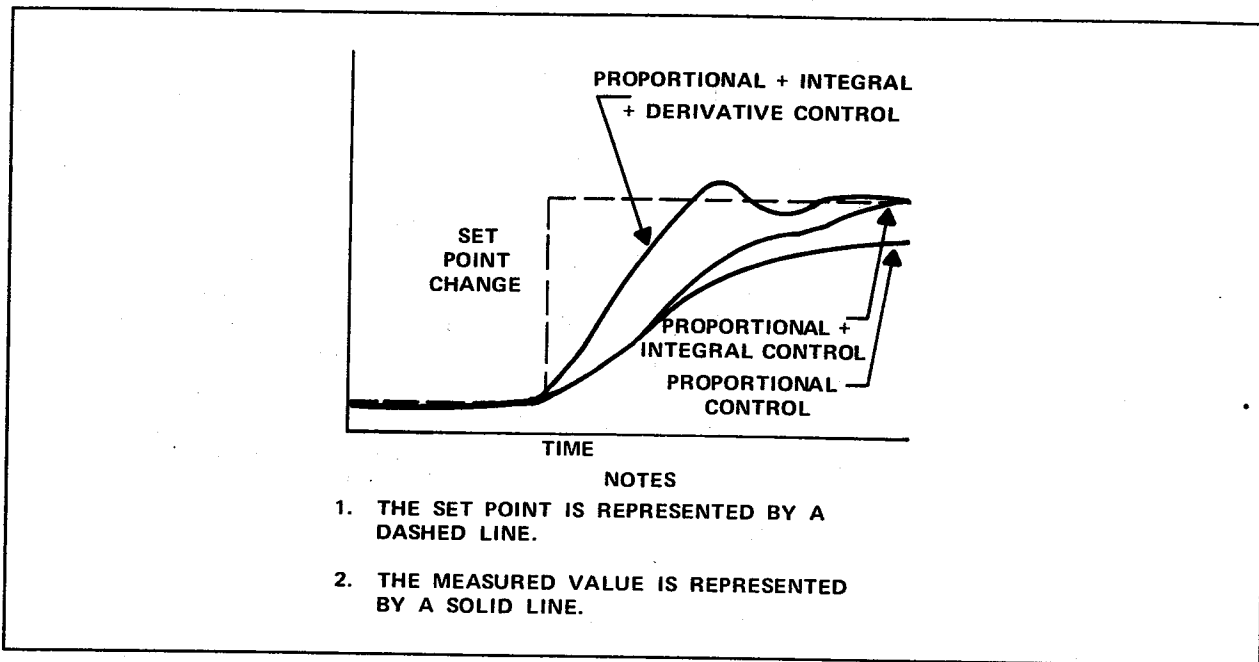


Figure 12. Effects of PID Control

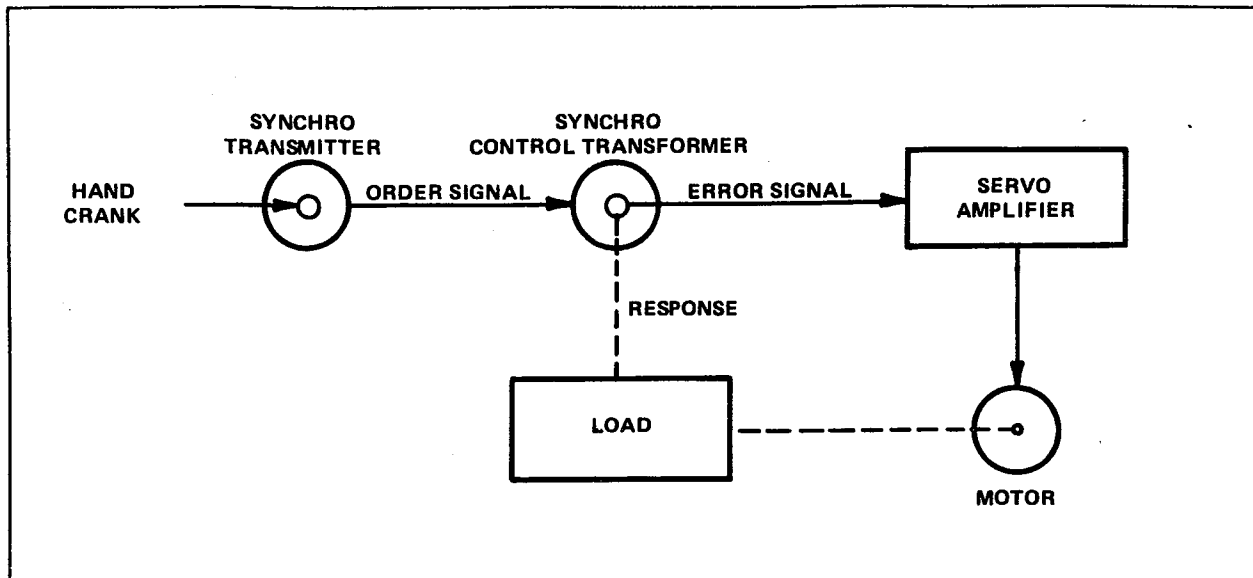


Figure 13. PD Control

## CLOSED-LOOP CHARACTERISTICS

### REACTION DELAY

In selecting a mode of process control, it is necessary to consider not only control system characteristics, but also the reaction of the process itself. Most processes require a certain amount of time to fully respond to a change in input. This amount of time is called process reaction delay and must be considered when selecting a mode of control. Another factor that must be considered is "transfer lag", a condition in which the corrective action must be transferred through one (or more) other media before affecting the process variable. The difference between process reaction delay and transfer lag is in the manner of application of correction to the process. All systems have some form of process reaction delay, but systems in which corrective action is transferred from medium to medium before application to the process also suffer additional transfer lag.

Process reaction delay is easily predictable; transfer lag is not. Long transfer lags constitute a difficult control problem. An additional, and more difficult, problem is that of transportation lag or dead time (or combination of both). This situation occurs when nothing occurs for some period of time after application of a correction. This situation results from delays due to the requirement that some part of the process may be physically transported from one place to another (hence "transportation" lag). In geared mechanical systems, this is known as "lost motion" (i.e., the amount of motion required to take up the slack in a gear train). All these factors must be considered in selecting a control mode. Table 1 is helpful in selecting a control mode.

**TUNING**

Three characteristics must be adjusted (tuned) in the controller for proper system operation:

- Proportional band
- Reset rate (reciprocal of integral time constant)
- Derivative control rate

These adjustments depend on the process and the mode of control. In a PI control system, the tuning of the system is dependent on the process characteristics shown in Table 2.

Reaction rate is the opposite of reaction delay. A short reaction delay equates to a fast reaction rate. A PD control system must have a moderate to long reaction delay and a narrow proportional band, because there is no

integral control to take care of the offset. The narrow proportional band is subject to overshoot and cycling, but the derivative portion of the controller tends to overreact whenever the error is changing rapidly, thus stabilizing the system within the proportional band.

**TABLE 1. CONTROL MODES**

Control Mode	Process Reaction Delay (Minimum)	Transfer Lag (Maximum)	Dead Time (Maximum)	Size of Load Upset (Maximum)	Speed of Load Upset (Maximum)
Proportional	Long or moderate (cannot be short)	Short or moderate	Short or moderate	Small	Slow
Proportional-plus-Integral (PI)	Any	Short or moderate	Short or moderate	Any	Slow
Proportional-plus-Derivative (PD)	Long or moderate (cannot be short)	Short or moderate	Short or moderate	Small	Any
Proportional-plus-Integral-plus-Derivative (PID)	Any	Any	Any	Any	Any

TABLE 2. CONTROLLER TUNED REACTION RATES

Process Characteristic		Controller Tuning	
Reaction Rate	Total Lag	Proportional Band	Reset Rate (Reciprocal of Integral Time Constant)
Slow	Short	Narrow	Fast
Slow	Moderate	Medium	Slow
Fast	Short	Medium	Fast
Fast	Moderate	Wide	Slow

When long transfer lags and/or long dead times are present, PID control is nearly always the only successful scheme of control. In PID control, the proportional band is very wide such that only a small portion of corrective action after an upset is due to proportional control. Most immediate reaction is due to the derivative portion of the controller so that when the error stops growing, derivative action disappears, leaving only the small signal. Due to proportional action, the controller senses recovery, and the integral portion of the controller repositions the system back to the original set point.

## FEATURES OF LC IN A PROGRAMMABLE CONTROLLER

### ALARMS

In closed-loop systems, alarms are generally used on the process variable and are either fixed points within the operational span of the system (see Figure 14) or are in a band about, and moving with, the set point (see Figure 15). Fixed alarms are called the high and low limit alarms; alarms moving with the set point are called deviation alarms. Deviation alarms apply equally to either side of the side of the set point; there are two low-deviation alarms (one on either side of the set point), and two high-deviation alarms (one on either side of the set point, and a greater distance from the set point than the low-deviation alarm). Each alarm of either type has an associated programmable deadband on the side toward the desired operating range. The deadband is used to prevent alarm "chatter".

The high alarm comes ON when the process variable exceeds the high alarm limit and goes OFF when the process variable is less than the high limit deadband. Further, high and low alarm limits, unlike deviation alarms, do not change with set point.

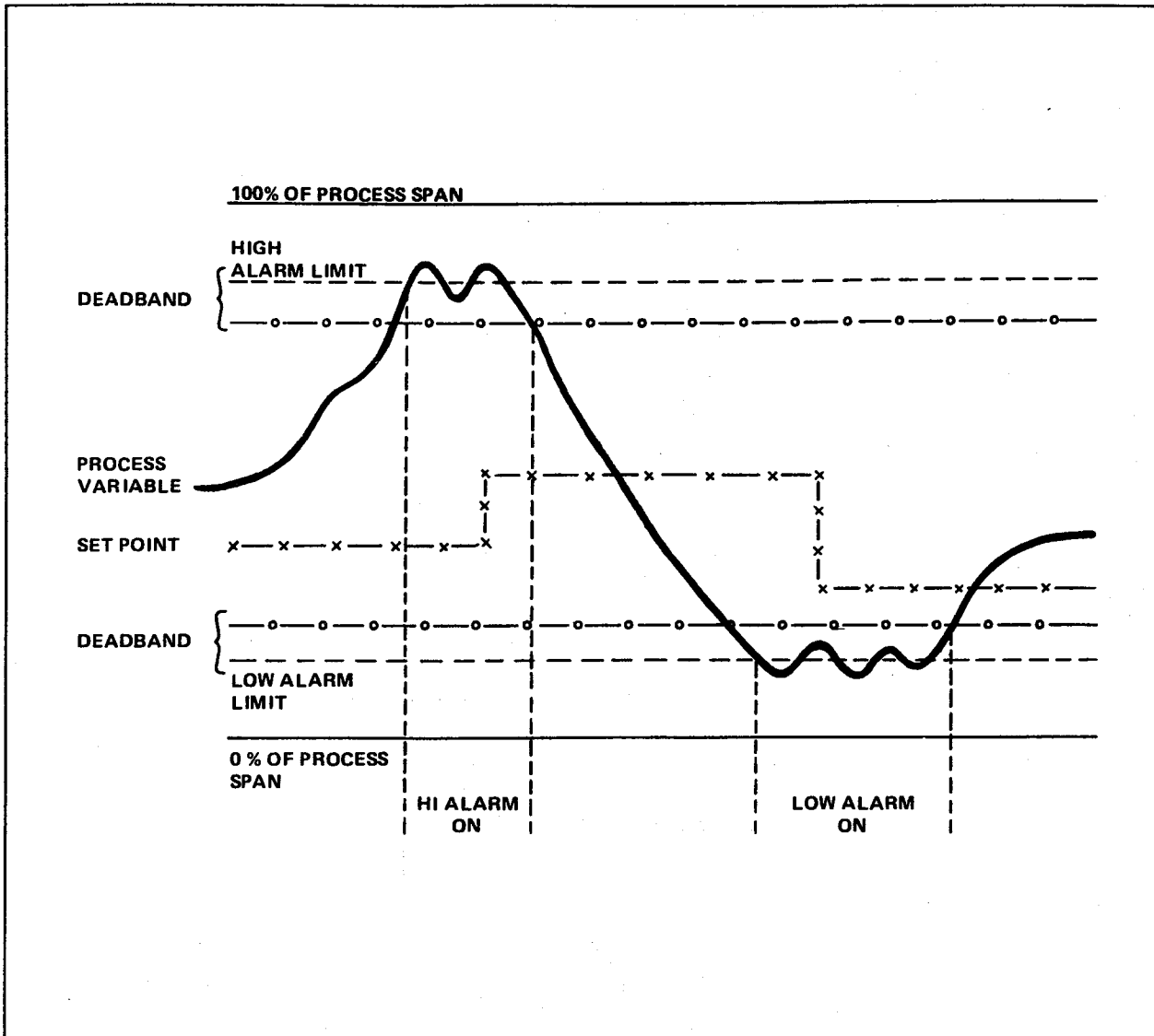


Figure 14. High and Low Limit Alarms

**BATCH UNIT**

A batch unit is used to start batch processes automatically. Without batch limits, starting a process from a cold start subjects the system to excessive error signals with a resultant large overshoot in the process variable.

Batch control establishes both a minimum output and a maximum output such that no matter how high the output may try to go, it is limited to an acceptable value by the batch unit high limit. Likewise, the batch unit preload ensures that the output cannot fall below a preload value. This feature provides rapid stabilization of batch processes. Batch limits are also used on the PC-1100 to allow user selection of 8-bit D-A converters without unnecessarily winding up the integral (reset) term. (See Figure 16). The PC-1200 allows 12 bit D-A converters.

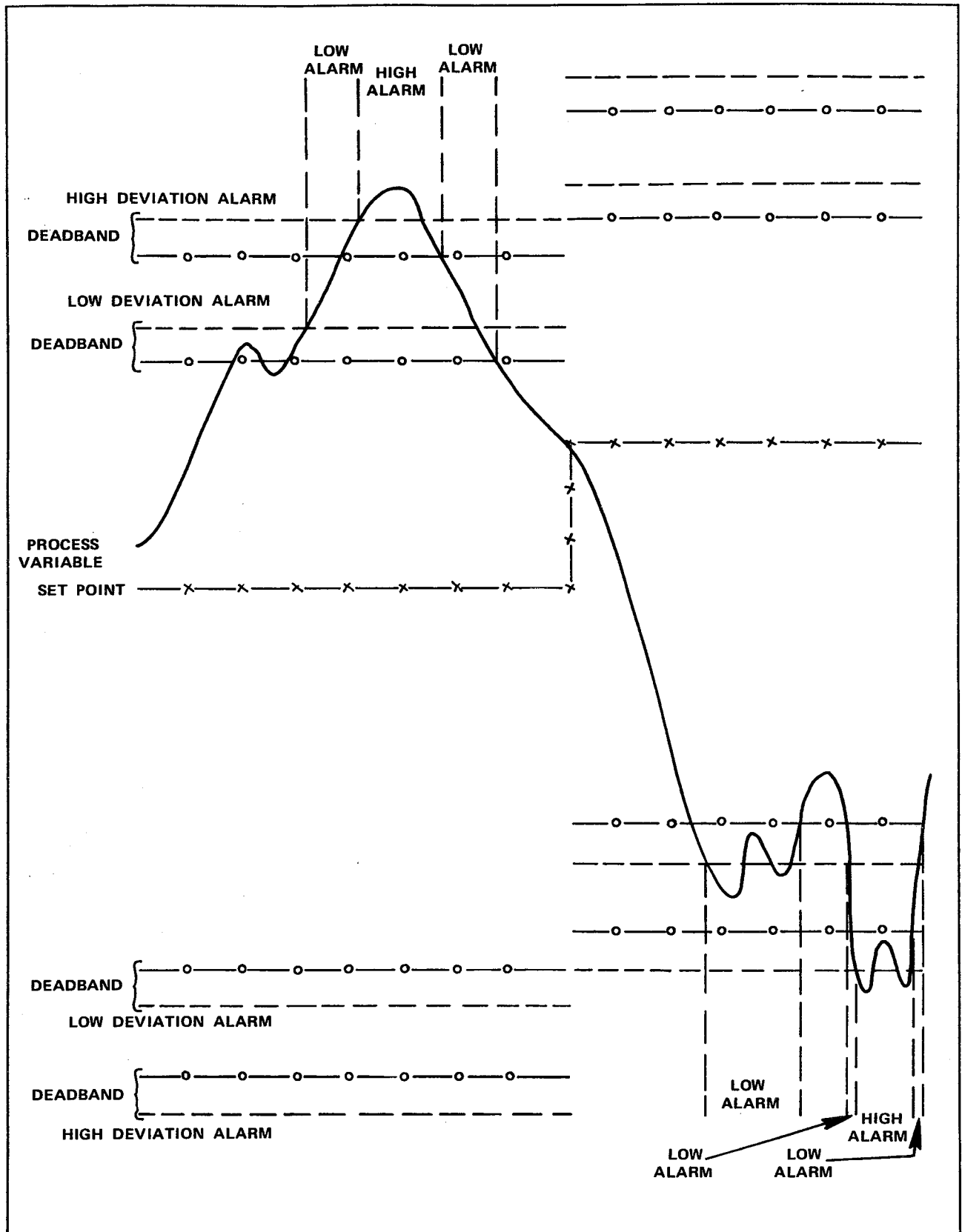


Figure 15. Deviation Alarms

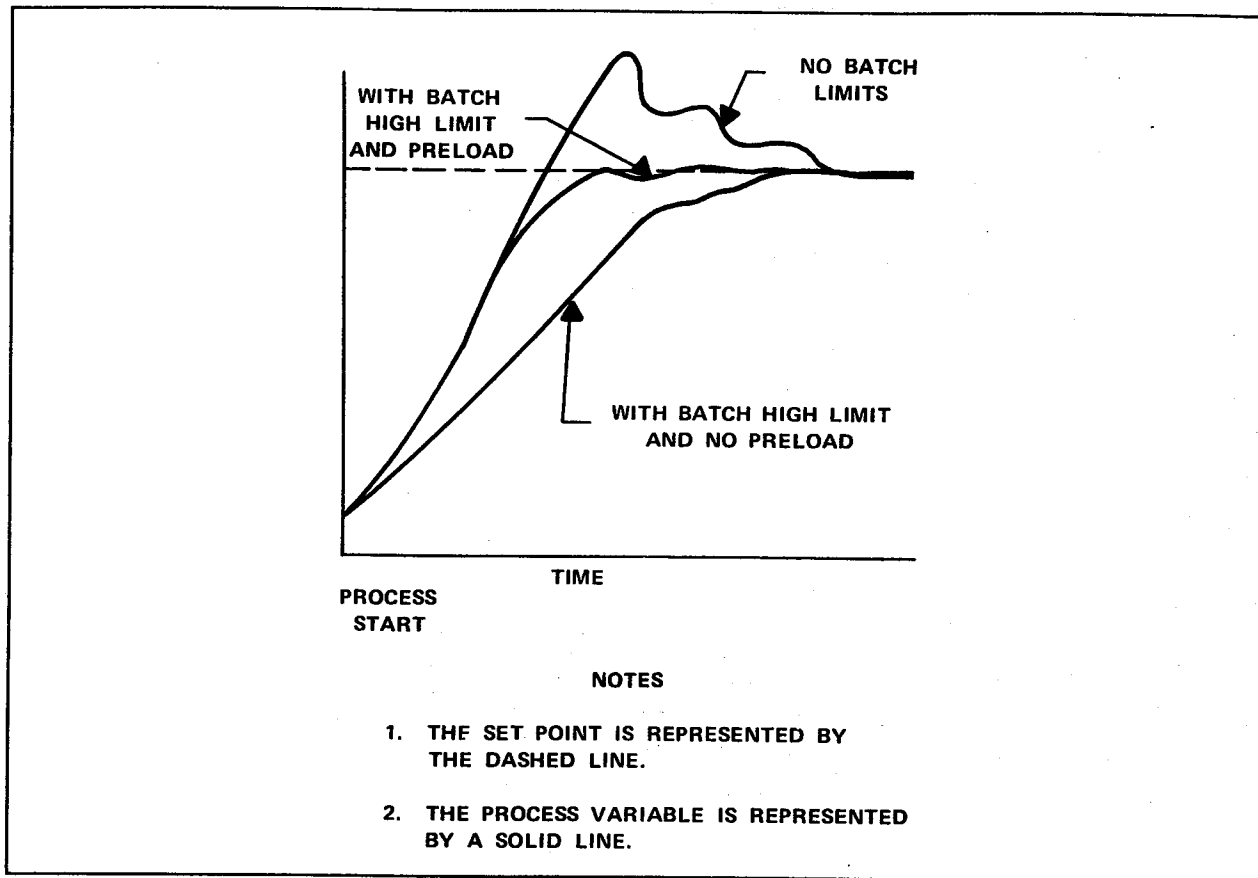


Figure 16. Batch Limit Effects

## ERROR DEADBAND

Error deadband control is used to eliminate response to small error values. A deadband is established about the set point, in which error values have no effect and are treated as zero error. The PC-1100 creates this deadband based on the low-deviation limit. The PC-1200 uses holding register HRXXXX-19 to define the error deadband. Beyond the deadband limits, the deadband value is subtracted from the actual error to get a "computed error", which is actually used in loop calculations. Figure 17 shows the effects of error deadband.

An error signal is computed only when the actual error signal exceeds the deadband values. The apparent result is no response to small errors and reduced response to larger errors. This is shown in Figure 17. PV represents the value of the process variable. Whenever, that value exceeds the deadband, E, the computed error signal is generated.

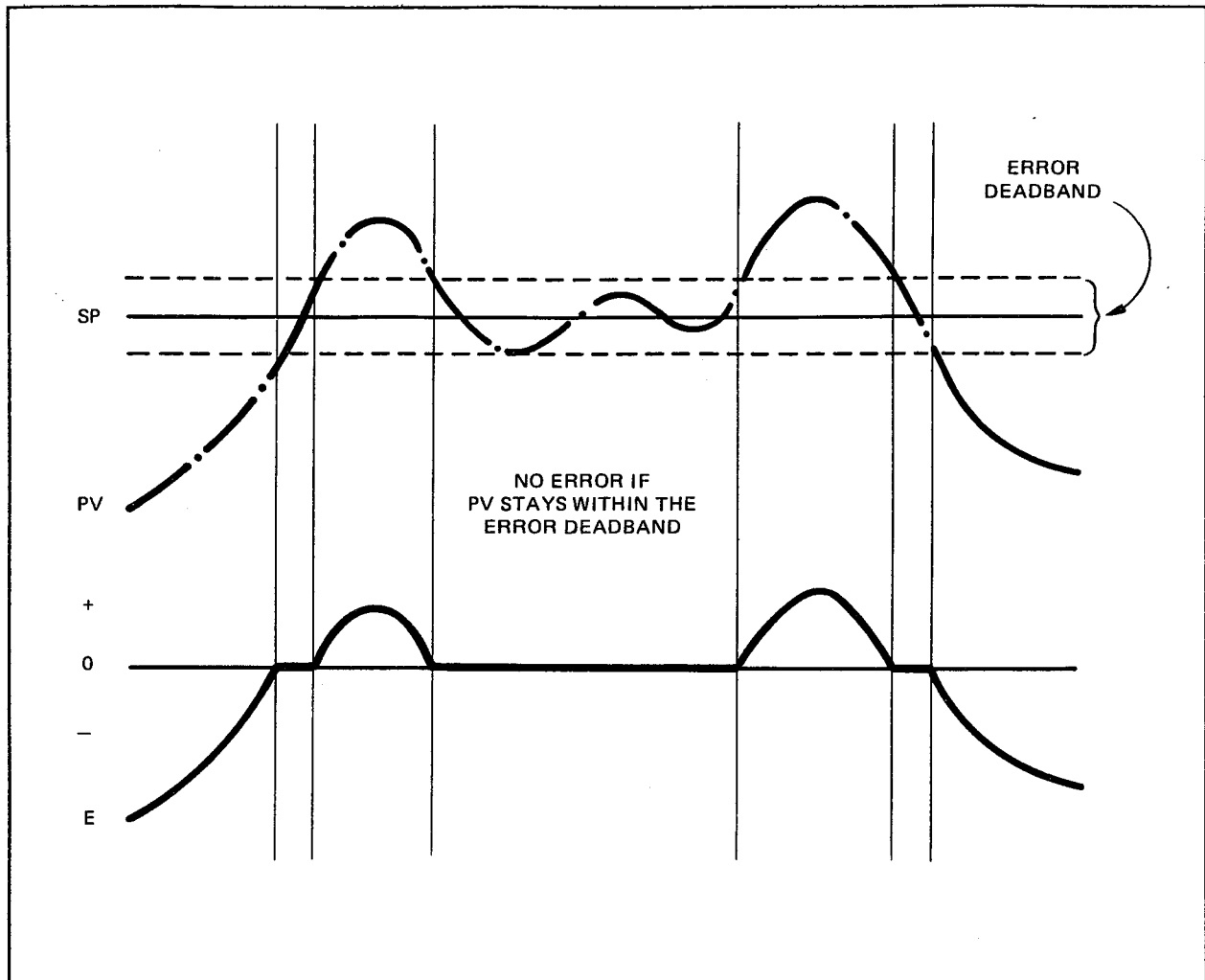


Figure 17. Effect of Error Deadband

### SAMPLE TIME

Process control is done in an interactive fashion. Of critical importance to the user are the maximum and minimum times between samples (loop calculations). The Numa-Logic LC function has a programmable sample time of 0.1 second to a maximum of 3276.7 seconds in 0.1 second increments. The user generally selects sample time as a function of the actual system response times.

### ANTI-RESET WINDUP

The integral term of a closed-loop system tries to continue increasing the loop output as long as there is a deviation from the set point. If that deviation cannot be eliminated, or if the deviation persists for a long period of time, the integral error accumulates, causing the output of the loop to maximize and remain maximized.



Once the output has maximized, there is no use in continuing to increase the contribution of the integral term. Conversely, once the output is zero, it is of no use to continue decreasing (increasing in a negative direction) the integral term. This is known as reset windup and is generally undesirable. The LC function has programmable batch unit high limit and preload values. If the output is outside these limits, the LC limits the output to the appropriate batch limit, then back calculates the reset term so that the batch unit high limit or preload is not exceeded. This is anti-reset windup. Anti-reset windup is also used to accommodate slew-limiting, described in the following paragraph.

### **SLEW-LIMITING**

It may be desirable to limit the rate at which the controller output may change (slew). This feature is called slew-limiting and prevents the rate of change of the output from exceeding preset limits. (See Anti-Reset Windup.) The slew-limiting term represents the maximum acceptable change in controller output in any given sample, as specified by the sample timer.

### **REMOTE SET POINT**

Generally, the set point for a system is entered from an operator station; however, sometimes it is necessary to accept a set point from another device. Conventional controllers require the purchase of a remote set-point option. Since the Numa-Logic Programmable Controller can accept a set point from an internal or external register source, this can be accomplished without such an option. The user can either program the programmable controller to perform the operations of the external device (for example, a cam timer) or condition the signal of an external device to provide a satisfactory input to the loop. This also facilitates cascading of loops, where the output of one loop (the outer loop) feeds the set point of another loop (the inner loop).

### **CASCADE**

The Cascade mode of operation is used when it is desired to control one loop with the output of another loop. Figure 18 shows a cascaded closed-loop system. In this system, hot water is heated by using steam. The system operates as follows:

1. A decrease in water temperature causes the output of the temperature controller to indicate a need for an increase in temperature.
2. This output is the set point for the flow controller and results in an increase in the set point, indicating the need for more steam flow.
3. The flow controller increases the amount of steam until the process variable (flow) indicates that the need is satisfied, which also means that the water temperature is high enough to satisfy the set point for the temperature controller.

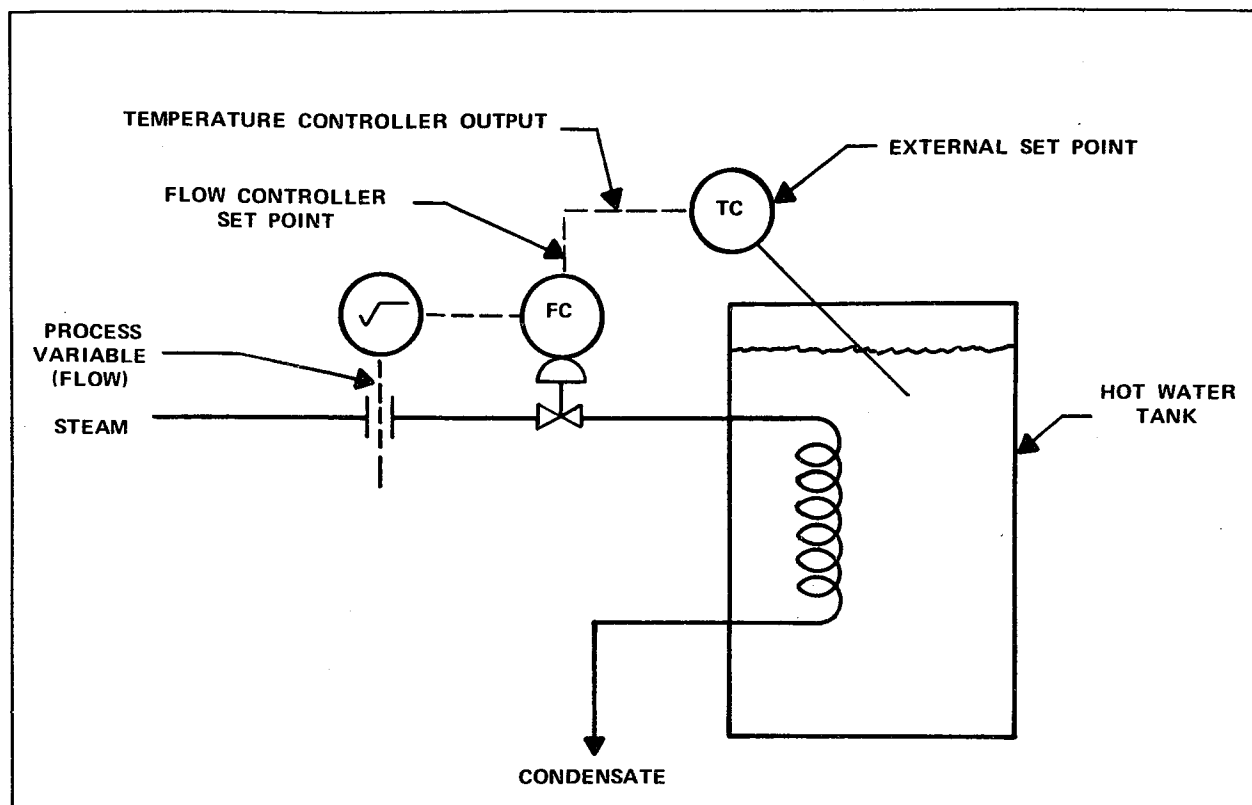


Figure 18. Cascaded Systems

The use of cascaded loops allows the flow controller to compensate for changes in steam pressure before that change affects the actual temperature, making a much tighter control system. A temperature controller alone would have compensated for steam pressure fluctuations, but only after it sensed a change in temperature.

### BUMPLESS TRANSFER

The term "bumpless transfer" means that transfers for the Auto mode to Manual (or vice versa) and setting up cascaded loops are accomplished without system upset. This is done by using an internal auto bias system such that when a transfer is made from one mode to another, no changes in system output are made. For example, during the transition from the Manual to Auto mode, the output is held constant; the integral term is reset; and  $SP_n$  (the set point in this sample) is set equal to  $PV_n$  (the process variable in this sample) to zero the error. Finally, the bias term is set equal to the output value (this is known as auto bias). This "balances" the PID equation such that no "bump" occurs.

## LOOP CONTROL WITH A PROGRAMMABLE CONTROLLER

### DESCRIPTION

The preceding general discussion of closed-loop control systems revolves around the general block diagram shown in Figure 5. When using a programmable controller, Figure 5 is modified as shown in Figure 19.

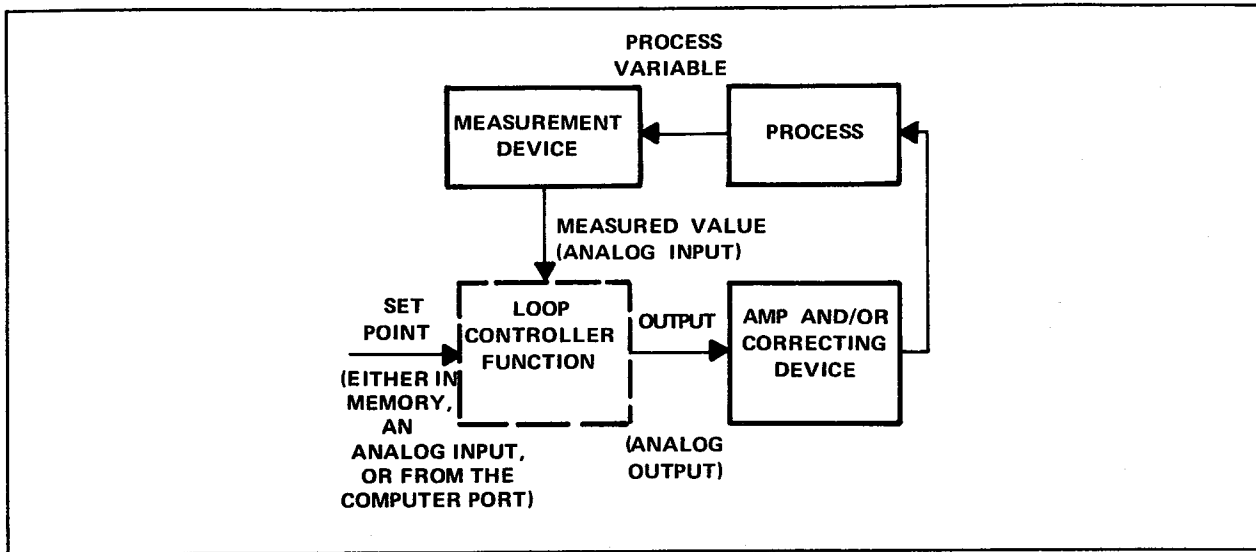


Figure 19. Closed-Loop Control with a Programmable Controller

System development is reduced to:

- Selecting the appropriate transducers, transmitters, process operators, etc.
- Selecting the control mode.
- Setting the system limits.
- Tuning the system.

The LC function is programmed as a special function and is very memory efficient (i.e., the loop table uses 36 words; the LC function uses six words; and the control circuit uses a minimum of four words). (See Table 3.) The Westinghouse Programmable Controllers can be used effectively in process control by coupling this efficiency with a wide range of analog and register modules. The symbols and basic concepts involved with the LC function are shown in Figure 20. An LC data worksheet is shown in Table 4. (A blank data worksheet is also provided at the end of this document, before the customer comment card. Copy as needed.)

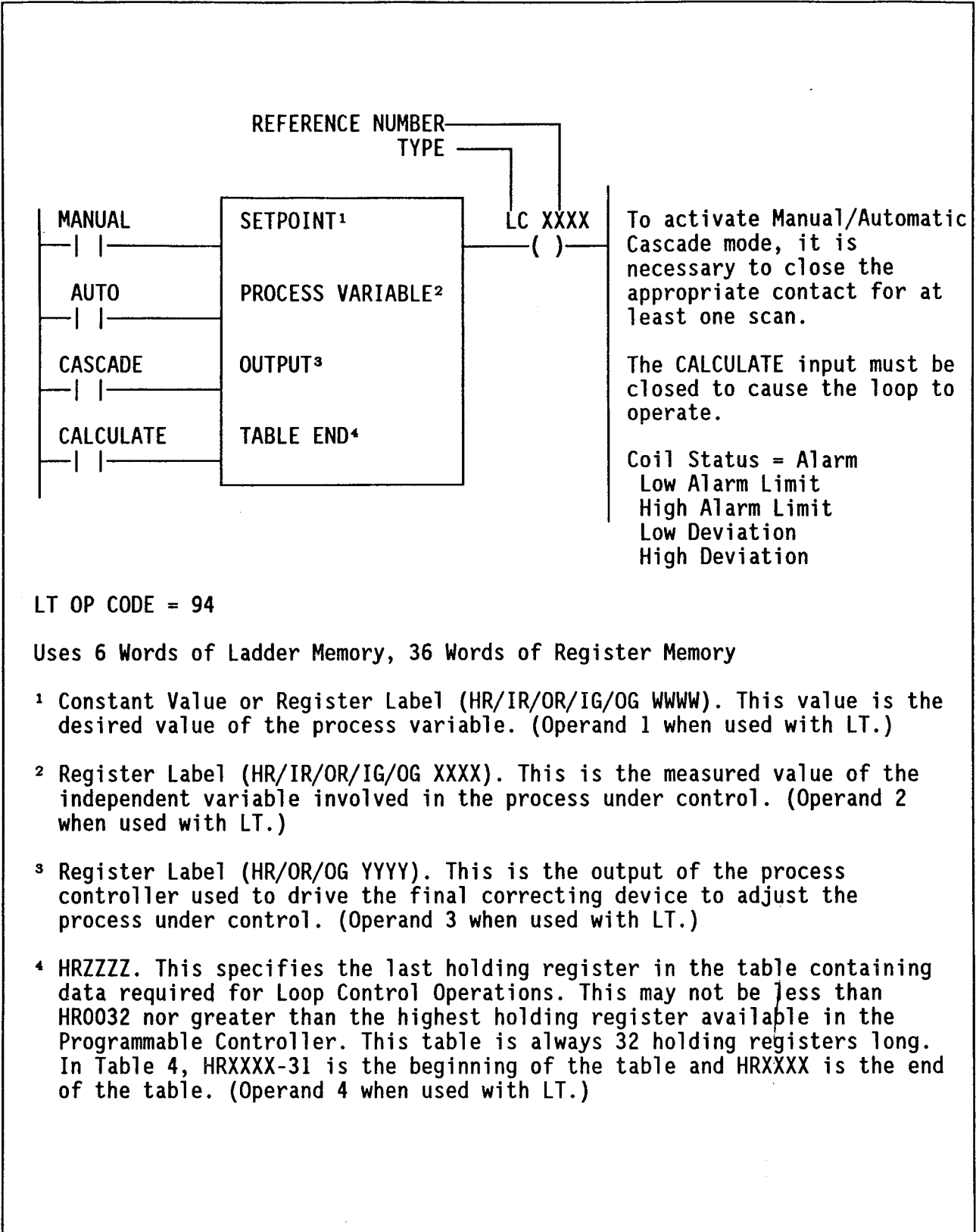


Figure 20. LC Function

TABLE 3. OUTPUT STATUS WORD (HRXXXX)

Bit Number	Definition	Bit Number	Definition
1	1 - High alarm limit active	9	1 - Record Auto/Manual
2	1 - Low alarm limit active	10	1 - Record Cascade
3	1 - High-deviation alarm active	11	1 - Illegal request-wrong mode
4	1 - Low-deviation alarm active	12 <sup>1</sup>	1 - An intermediate term is out of range (PC-1200 only)
5	1 = Sign of error-negative 0 = Sign of error-positive	13	1 = The calculated output is out of range (PC-1200 only)
6	1 = Positive slew-limiting	14 <sup>1</sup>	1 = A user supplied term would result in Divide by zero (PC-1200 only)
7	1 = Negative slew-limiting	15	Reserved for future use
8 <sup>1</sup>	1 = Process variable or set point variable is out of range (PC-1200 only)	16	Reserved for future use

<sup>1</sup>These error bits are latched in the OSW and must be cleared by the user. The coil will remain energized as long as any of these bits are set.

TABLE 4. LC DATA WORKSHEET

Loop Title:		C	H	I	O	O	Register Type & No. (Data if CV)	Comment
	Set Point	*	*	*	*	*		
	Process Variable	*	*	*	*	*		
	Output	*	*	*	*	*		
Loop Coll #:	Loop Table End	*	*	*	*	*		

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity	Value/Remarks	
			PC-1100	PC-1200
HRXXX-35		Reserved	C	
HRXXX-34		Derivative Coefficient	C	
HRXXX-33		Previous Integral Term	C	
HRXXX-32		Integral Coefficient	C	
HRXXX-31		Proportional Term ( $\pm 32,767$ )	C	
HRXXX-30		Integral Term ( $\pm 32,767$ )	C	
HRXXX-29		Derivative Term ( $\pm 32,767$ )	C	
HRXXX-28		SP <sub>n</sub> - Set Point This Sample	C	
HRXXX-27		PV <sub>n</sub> - Process Variable This Sample	C	
HRXXX-26		Time Counter - Elapsed Sample Time	C	
HRXXX-25		SP <sub>n-1</sub> - Set Point Previous Sample	C	
HRXXX-24		PV <sub>n-1</sub> - Process Variable Previous Sample	C	
HRXXX-23		E <sub>n-1</sub> - Error Previous Sample*	C	
HRXXX-22		Bias (0 to Maximum Output)	C	
HRXXX-21		Functional Gain	C	
HRXXX-20		Configuration Input Word (See Below)	U	
HRXXX-19		Error Deadband (0-9999)	PC-1200 Only U	
HRXXX-18		Derivative Decay Coefficient (0-99.99%)	PC-1200 Only U	
HRXXX-17		Integral Sum ( $\pm 32,767$ )	C	
HRXXX-16		E <sub>n</sub> - Error This Sample*	C	
HRXXX-15		T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	
HRXXX-14		T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	
HRXXX-13		T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	
HRXXX-12		K <sub>c</sub> - Proportional Gain (0 - 99.99)	U	
HRXXX-11		Inner Loop Pointer (Loop Table End)	U	
HRXXX-10		Outer Loop Pointer (Loop Table End)	U	
HRXXX-9		Alarm Deadband (0 - Max PV)	U	
HRXXX-8		Batch Unit Preload (0 - Max Output)	U	
HRXXX-7		Batch Unit Hi Limit (0 - Max Output)	U	
HRXXX-6		Neg. Slew Limit (Max - $\Delta$ Output/Sample)	U	
HRXXX-5		Pos. Slew Limit (Max + $\Delta$ Output/Sample)	U	
HRXXX-4		Low Deviation Alarm Limit (0 - Max PV)	U	
HRXXX-3		High Deviation Alarm Limit (0 - Max PV)	U	
HRXXX-2		Low Alarm Limit (0 - Max PV)	U	
HRXXX-1		High Alarm Limit (0 - Max PV)	U	
HRXXX		Output Status Word	C	

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXX-20)

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected		9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	
2	1 = Integral Mode Selected		10	1 = Batch Unit Selected	
3	1 = Derivative Mode Selected		11	RESERVED FOR CONTROLLER USE	
4	1 = Deviation Alarms Selected		12	RESERVED FOR FUTURE USE	
5	1 = Error Deadband Selected		13	RESERVED FOR FUTURE USE	
6	RESERVED FOR FUTURE USE		14	PC-1100: Coefficient Lock	
7	1 = Slew Limiting Selected		15	RESERVED FOR FUTURE USE	
8	1 = Reverse Action Selected 0 = Direct Action Selected		16	RESERVED FOR FUTURE USE	

Table 4

# LC

## LC FEATURES

The LC function has the following features:

- Cascaded loops
- Remote set point
- Auto/Manual/Cascade control with bumpless transfer
- Direct or reverse acting
- Auto bias
- Anti-reset windup
- Slew-limiting
- High and low alarms
- Deviation alarms
- Error deadband
- Calculate dpv/dt or de/dt

## LC OPERATION

The LC function has two methods of operation:

- Calculate circuit not conducting

Alarms are processed if selected and enabled. The Manual mode is functional. Auto or Cascade modes are not calculated, but the timer is functional. On time out, the output is not updated until the calculated line is closed (conducting).

- Calculate circuit conducting

Loop control is fully functional. If necessary, the calculate line can be used to stagger the calculation of several loops; this prevents the possibility of several loops being calculated during a given scan.

Mode selection is accomplished by momentarily energizing either the Manual, Auto, or Cascade request lines. This sets the appropriate Auto/Manual or Cascade bit in the Output/Status Word. If more than one request line is conducting during a given scan, the mode is selected as follows:

1. In all cases, the Manual contact takes precedence.
2. If an illegal closure is made without Manual involvement, the Last Valid State is maintained.
3. Allowable transitions are shown in Table 5.

**TABLE 5. MANUAL LOOPS**

START - BOTH LOOPS IN MANUAL	
Inner Loop	Outer Loop
<ol style="list-style-type: none"> <li>1. Adjust the output until it is near desired point.</li> <li>2. Set the inner loop to Auto.</li> <li>3. Set the inner loop to Cascade.</li> </ol>	<ol style="list-style-type: none"> <li>1. Load the desired set point.</li> <li>2. Set point is passed from the inner loop's output</li> <li>3. Set the output loop to Auto.</li> </ol>
<p>Notes</p> <ol style="list-style-type: none"> <li>1. If the inner loop goes from Cascade to Auto, the outer loop goes from Auto to Manual automatically.</li> <li>2. Loop controllers will not go into Cascade operation unless the inner and outer loop pointers are in the loop table.</li> </ol>	

### LC CONFIGURATION

Loops are configured by first programming the LC special function, and then entering data into the loop table.



# LC

The LC configurations include:

- Bumpless transfer
- Anti-reset windup
- Coefficient Lock
- Slew-limiting
- Alarms
- Reverse-acting
- $dpv/dt$  vs.  $de/dt$
- Cascaded loops
- PID selection

## Bumpless Transfer

There are four types of bumpless transfer:

- Manual to Auto

During the transition, the output is held constant; the integral term is reset; and  $SP_n$  (the set point in this sample) is set equal to  $PV_n$  (the process variable in this sample), zeroing the error. The bias term is set equal to the output value.

- Auto Manual

This is the same as for Manual to Auto. However, no further changes in output occur until acted upon by the operator.

- Auto Cascade

PC-1100 - The current value of the output is transferred to the register location where the outer loop set point resides.

PC-1200 - Outer loop goes from Auto to Manual.

- Cascade to Auto

The set-point value is retained until it is manually changed.

### Anti-Reset Windup

Two limits are programmed by the user to tailor anti-reset windup to the particular system being controlled:

- Batch unit high limit

This is the maximum possible value of loop output.

- Batch unit preload

This is the minimum possible value of loop output.

If either limit is reached, the value  $E_i$  is back-calculated to maintain that output. This prevents the reset term from accumulating further error and saturating,, or "winding up".

If anti-reset windup is not selected, the processor defaults to 255 for batch unit high limit and zero for batch unit preload.

### Coefficient Lock (PC-1100 Only)

The four least significant registers in the Loop Table are used for intermediate math calculations. These calculations, the Derivative Coefficient and Derivative Exponent and Integral Coefficient and Integral Exponent are based upon the tuning parameters  $T_d$ ,  $T_i$ ,  $T_s$ , and  $K_c$ . When tuning is complete and no further adjustments are made to the parameters, the values in the four least significant registers will not change. Therefore, it is not necessary to recalculate these values during each processor scan. Bypassing these calculations will improve the Loop Controller execution time. To do so, set the Coefficient Lock (Bit 14) in the Configuration Input Word to 1.

### Alarms

High alarm occurs when the process variable exceeds a preset high alarm limit. Low alarm occurs when the process variable is below a preset low alarm limit.

High-deviation alarm occurs if the process variable differs from the set point by a selectable high-deviation alarm limit. Low-deviation alarm is similar to high-deviation alarm, but has a (smaller) selectable difference.

1. Deviation alarms are not calculated while in the Manual mode. Low-deviation alarm limit locations in the loop table are used to specify error deadband values (if selected).
2. If a high, low, high-deviation, or low-deviation alarm is active, the coil turns ON to signify an alarm.
3. A small deadband should be programmed for all alarm limits to eliminate alarm oscillation. The user should set the alarm deadband values to be consistent with his process needs.

# LC

## Reverse-Acting

Selection of reverse-acting causes the calculation of  $e = PV - SP$ , instead of  $e = SP - PV$ .

$dpv/dt$  vs.  $de/dt$

If set-point changes adversely affect the derivative term, selection of  $dpv/dt$  eliminates the effect of rapid set-point changes on the derivative term.

## Cascaded Loops

The inner loop pointer (loop table end for the inner loop) and the outer loop pointer (loop table end for the outer loop) must be entered by the user. These pointers are used by the function to check permissible Auto/Manual/Cascade transfers and to pass set-point information between loops.

## PID Selection

P, I, and D are independently selectable as part of the Input Configuration Word. This allows the selection of any PID combination; it also allows the user to change modes in an online fashion, while controlling a process.

## LC APPLICATIONS

The examples in this description are based on the PC-1100 and eight bit resolution input output cards, i.e., NL-1050 and NL-1057. The PC-1100 LC special function is restricted to inputs and outputs within this range while the PC-1200 is capable of handling inputs and outputs in the range of 0 through 9999. The PC-1200 was designed to accept new analog cards with higher resolution. The following table summarizes the number of inputs and outputs are available with various input and output card resolutions.

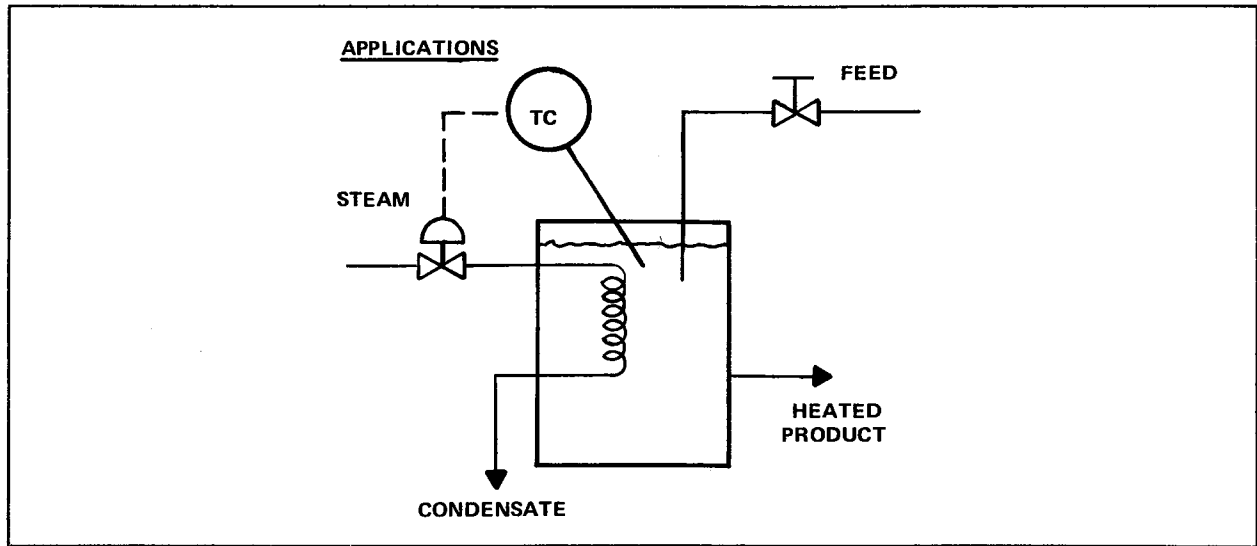
<u>Card Resolution (bits)</u>	<u>Maximum Range (Decimal)</u>
eight	255
ten	1023
twelve	4095
twelve + sign (13)	8192

If you want to make comparisons in the examples using higher resolutions, simply replace the eight bit range of 255 with the corresponding resolution range of choice in the calculations.

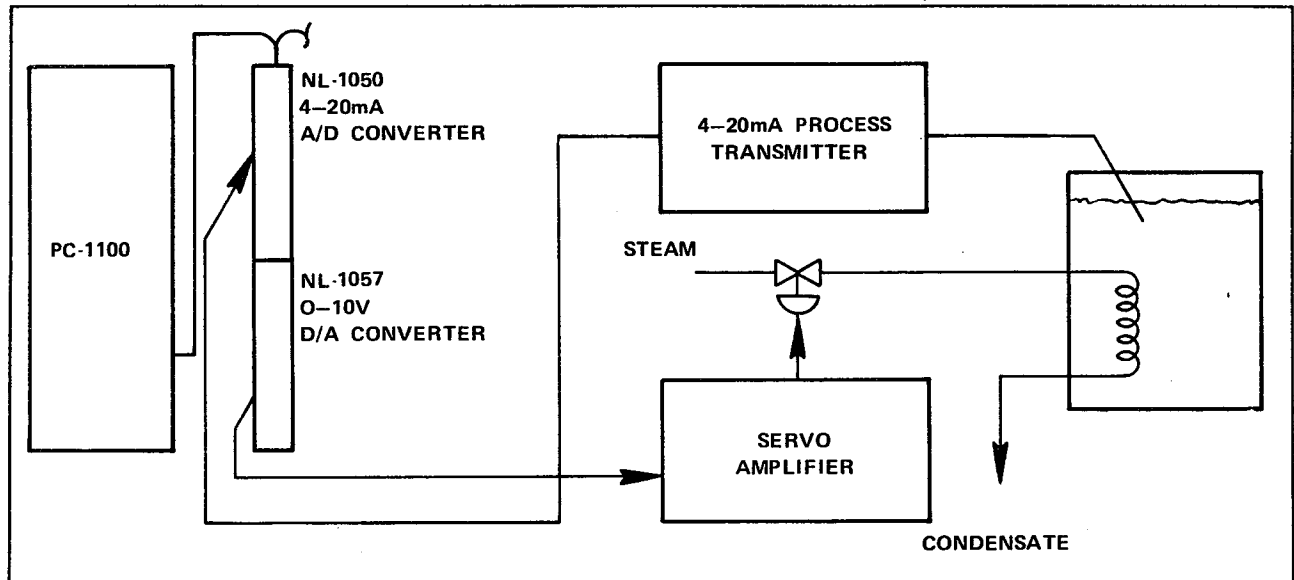
As higher resolution A/D input cards become available, it may be desirable to use D/A outputs of lower resolution. Control elements (e.g., pipe valve actuator) typically do not have control actuators which can express or should use higher resolution. Lower resolution on the output can act as a built-in error deadband, thus eliminating hunting or jittering on the control actuator.

**Example 1**

The system shown in Figure 21 uses steam to heat a liquid product. The temperature is maintained by a temperature control loop, which senses the temperature (process variable), compares it to the set point (error equals set point minus process variable), and through the temperature controller, changes the output to change the amount of steam allowed into the heating coil in the tank. The system operates in the proportional mode. Implementing this system with a programmable controller results in a system similar to that shown in Figure 22.



**Figure 21. Single-Loop Control System**



**Figure 22. Single-Loop Hardware Configuration**

## LC

Assume that the temperature probe has a range of measurement from 0°C through 250°C. The process transmitter sends a 4 through 20 mA signal to the NL-1050

Analog-to-Digital Converter. The PC-1100 A-D converter has an 8-bit resolution, dividing the 4 through 20 mA (0°C through 250°C) span into 255 parts. Thus:

$$\frac{255 \text{ units}}{250^\circ\text{C}} = \frac{1.02 \text{ units}}{^\circ\text{C}}$$

If the set point of the system is to be 100°C, then the following calculation yields the number that must be placed in the set point register corresponding to 100°C.

$$\frac{1.02 \text{ units}}{^\circ\text{C}} \times 100^\circ\text{C} = 102 \text{ units}$$

The LC function, by using the set point and measured value of the process variable, develops an output to an output register for conversion to an analog voltage to control the valve position.

Since the output ranges from 0 percent through 100 percent of valve opening, the amount that the steam valve is open at any given time can be calculated by the following process:

$$\frac{\text{Number in Output Register for NL-1057}}{255} \times 100\% = \text{percentage of valve opening}$$

For example:

$$\frac{133}{255} \times 100\% = 52.15 \text{ percent open}$$

The loop can now be programmed as shown in Figure 23.

In Table 6, the high alarm limit is set to 256 so that a high alarm will never be active for any output value.

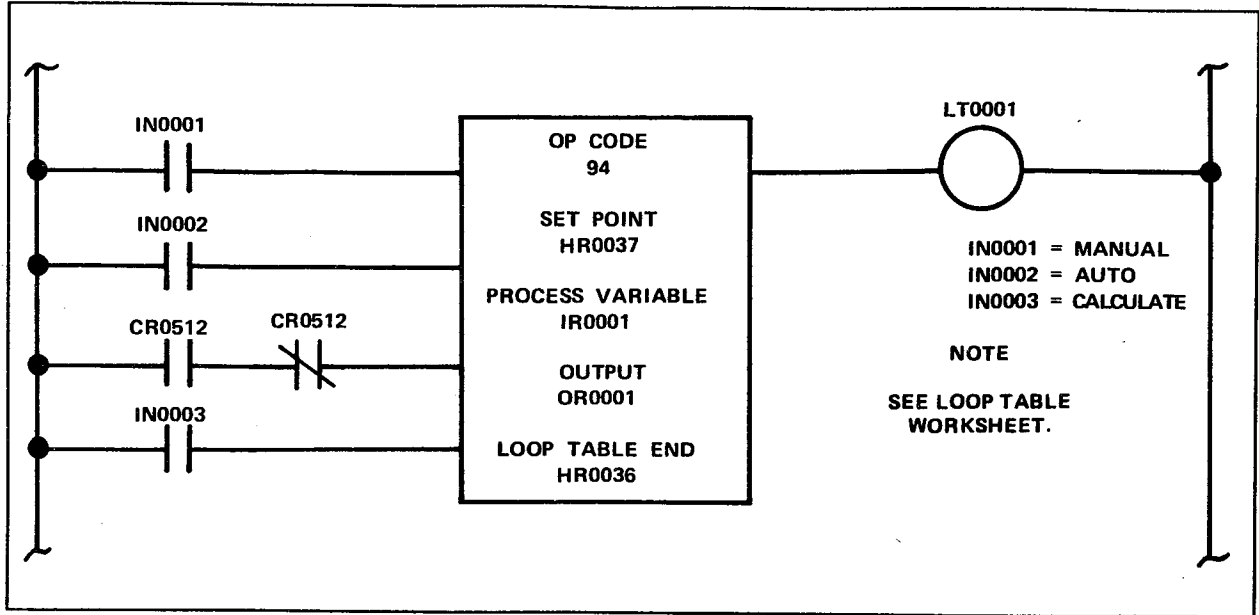


Figure 23. Loop Program

Initially, an arbitrary figure is used for proportional gain. This arbitrary figure allows the system to operate such that the process can be lined out in the Manual mode. Once lined out, a tuning technique, such as that described later in this section, can be used to properly tune the gain.

Figure 24 extends the example shown in Figure 21.

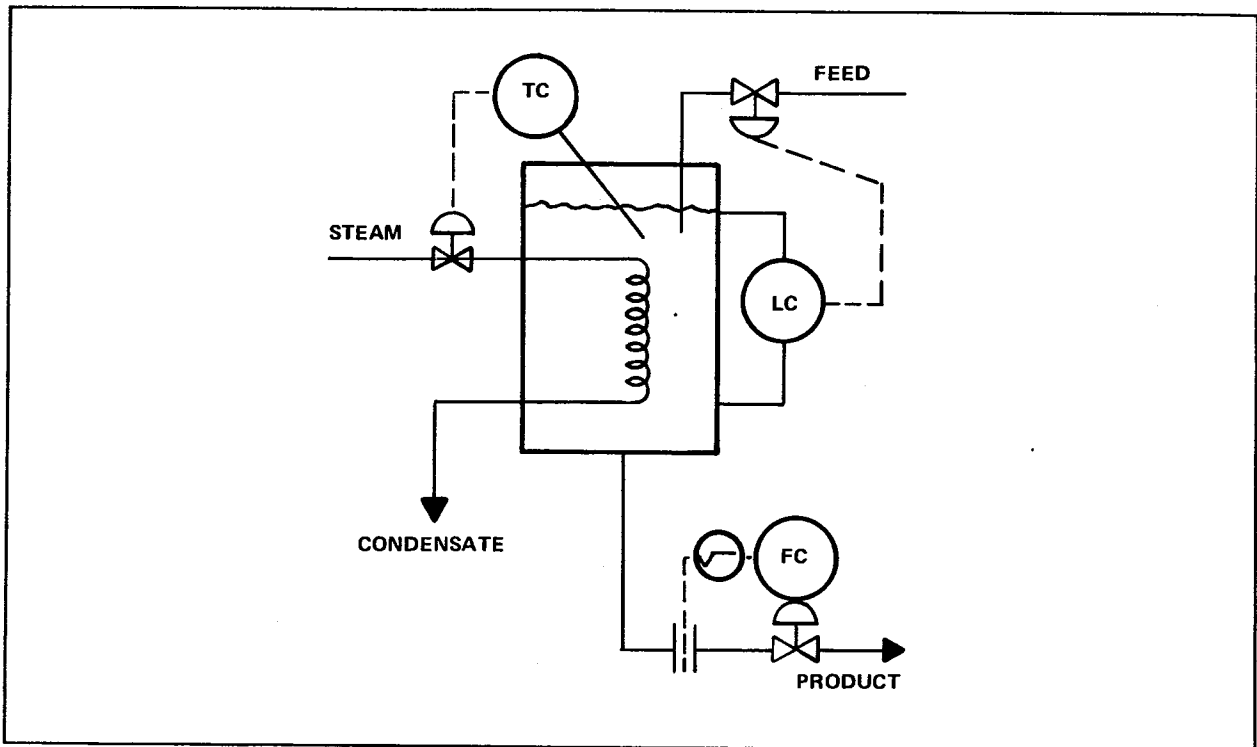


Figure 24. Three-Loop System

TABLE 6. TEMPERATURE CONTROL LOOP (LT0001) DATA WORKSHEET

Loop Title:		C	H	I	O	I	O	Register Type & No. (Data if CV)	Comment
Temperature Control Loop	Set Point	*	*	*	*	*	*	HR0037	Set point is placed in HR0037
	Process Variable		*	*	*	*	*	IR0001	using a program loader.
	Output		*	*	*	*	*	OR0001	
Loop Coil #: LT-0001	Loop Table End		*	*	*	*	*	HR0036	

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity	Value/Remarks	
			PC-1100	PC-1200
HRXXX-35	HR0001	Reserved	C	
HRXXX-34	HR0002	Derivative Coefficient	C	
HRXXX-33	HR0003	Previous Integral Term	C	
HRXXX-32	HR0004	Integral Coefficient	C	
HRXXX-31	HR0005	Proportional Term ( $\pm 32.767$ )	C	
HRXXX-30	HR0006	Integral Term ( $\pm 32.767$ )	C	
HRXXX-29	HR0007	Derivative Term ( $\pm 32.767$ )	C	
HRXXX-28	HR0008	SP <sub>n</sub> - Set Point This Sample	C	
HRXXX-27	HR0009	PV <sub>n</sub> - Process Variable This Sample	C	
HRXXX-26	HR0010	Time Counter - Elapsed Sample Time	C	
HRXXX-25	HR0011	SP <sub>n-1</sub> - Set Point Previous Sample	C	
HRXXX-24	HR0012	PV <sub>n-1</sub> - Process Variable Previous Sample	C	
HRXXX-23	HR0013	E <sub>n-1</sub> - Error Previous Sample*	C	
HRXXX-22	HR0014	Bias (0 to Maximum Output)	C	
HRXXX-21	HR0015	Functional Gain	C	
HRXXX-20	HR0016	Configuration Input Word (See Below)	U	0201H
HRXXX-19	HR0017	Error Deadband (0-9999)	U	PC-1200 Only
HRXXX-18	HR0018	Derivative Decay Coefficient (0-99.99%)	U	PC-1200 Only
HRXXX-17	HR0019	Integral Sum ( $\pm 32.767$ )	C	
HRXXX-16	HR0020	E <sub>n</sub> - Error This Sample*	C	
HRXXX-15	HR0021	T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	Not used in example
HRXXX-14	HR0022	T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	Not used in example
HRXXX-13	HR0023	T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	100 tenths of second
HRXXX-12	HR0024	K <sub>c</sub> - Proportional Gain (0 - 99.99)	U	5.0
HRXXX-11	HR0025	Inner Loop Pointer (Loop Table End)	U	Not used in example
HRXXX-10	HR0026	Outer Loop Pointer (Loop Table End)	U	Not used in example
HRXXX-9	HR0027	Alarm Deadband (0 - Max PV)	U	Not used in example
HRXXX-8	HR0028	Batch Unit Preload (0 - Max Output)	U	0000 Default
HRXXX-7	HR0029	Batch Unit Hi Limit (0 - Max Output)	U	255
HRXXX-6	HR0030	Neg. Slew Limit (Max - $\Delta$ Output/Sample)	U	Not used in example
HRXXX-5	HR0031	Pos. Slew Limit (Max + $\Delta$ Output/Sample)	U	Not used in example
HRXXX-4	HR0032	Low Deviation Alarm Limit (0 - Max PV)	U	Not used in example
HRXXX-3	HR0033	High Deviation Alarm Limit (0 - Max PV)	U	Not used in example
HRXXX-2	HR0034	Low Alarm Limit (0 - Max PV)	U	0000
HRXXX-1	HR0035	High Alarm Limit (0 - Max PV)	U	255
HRXXX	HR0036	Output Status Word	C	

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXX-20)      16 15 14 13      12 11 10 9      8 7 6 5      4 3 2 1      = 0201H

   0 0 0 0      0 0 1 0      0 0 0 0      0 0 0 1

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected	1	9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	0
2	1 = Integral Mode Selected	0	10	1 = Batch Unit Selected	1
3	1 = Derivative Mode Selected	0	11	RESERVED FOR CONTROLLER USE	0
4	1 = Deviation Alarms Selected	0	12	RESERVED FOR FUTURE USE	0
5	1 = Error Deadband Selected	0	13	RESERVED FOR FUTURE USE	0
6	RESERVED FOR FUTURE USE	0	14	PC-1100: Coefficient Lock	0
7	1 = Slew Limiting Selected	0	15	RESERVED FOR FUTURE USE	0
8	1 = Reverse Action Selected 0 = Direct Action Selected	0	16	RESERVED FOR FUTURE USE	0

Table 6

This extended system uses three independent loops to control the temperature, liquid level, and flow.

### Loop System

Standard 4 through 20 mA process transmitters are used. The parameters of the system are:

- Temperature measuring device  
(100°C through 300°C)
- Flow measuring device

This example assumes the use of an orifice differential pressure transducer. The square root of the transducer output is linear with the flow rate. ((01pm)<sup>2</sup> through (100 lpm)<sup>2</sup>)

- Level measuring device  
(0 cm through 200 cm)

Since these inputs will be applied to an NL-1045 A-D converter these input ranges must be converted to unit equivalents for use with the programmable controller. For the temperature loop:

$$\frac{255 \text{ units}}{300^\circ\text{C} - 100^\circ\text{C}} = \frac{255 \text{ units}}{200^\circ\text{C}} = \frac{1.275 \text{ units}}{^\circ\text{C}}$$

For the flow loop:

$$\frac{255 \text{ units}}{(100 \text{ lpm})^2} = \frac{0.0255 \text{ units}}{1 \text{ pm}^2}$$

#### Note

The processor will be required to take the square root of the flow input when the measuring device is an orifice/differential pressure transducer.

For the level control loop:

$$\frac{255 \text{ units}}{200 \text{ cm}} = \frac{1.275 \text{ units}}{\text{cm}} - \frac{1 \text{ units}}{\text{cm}}$$



The programmable controller configuration of the system is shown in Figure 25.

In addition to the measuring devices, the temperature loop and flow loop have deviation alarms, and the level control loop has high and low alarm limits.

In the temperature loop, the low-deviation alarm limit is  $\pm 10^{\circ}\text{C}$  (set point  $\pm 10^{\circ}\text{C}$ ), while the high-deviation alarm limit is  $\pm 20^{\circ}\text{C}$  (set point  $\pm 20^{\circ}\text{C}$ ). A deadband of  $2^{\circ}\text{C}$  will be set for these limits. The settings are:

- Low-deviation alarm limit

$$10^{\circ}\text{C} \times \frac{1.275 \text{ units}}{^{\circ}\text{C}} = 12.75 \text{ units}$$

~ 12 units

- High-deviation alarm limit.

$$20^{\circ}\text{C} \times \frac{1.275 \text{ units}}{^{\circ}\text{C}} = 25.50 \text{ units}$$

~ 25 units

- Alarm deadband

$$2^{\circ}\text{C} \times \frac{1.275 \text{ units}}{^{\circ}\text{C}} = 2.55 \text{ units}$$

~ 2 units

These values will be used in the loop table worksheet.

The flow loop will have a low-deviation alarm limit of  $\pm 20$  lpm (set point  $\pm 20$  lpm), a high-deviation alarm limit of  $\pm 40$  lpm (set point  $\pm 40$  lpm), and a deadband of 7 lpm. The settings are:

- Low-deviation alarm limit

$$\sqrt{\frac{0.0255 \text{ units}}{\text{lpm}^2}} \times 20 \text{ lpm} = 3.19 \text{ units}$$

~ 3 units

- High-deviation alarm limit

$$\sqrt{\frac{0.0255 \text{ units}}{\text{lpm}^2}} \times 40 \text{ lpm} = 6.38 \text{ units}$$

~ 6 units

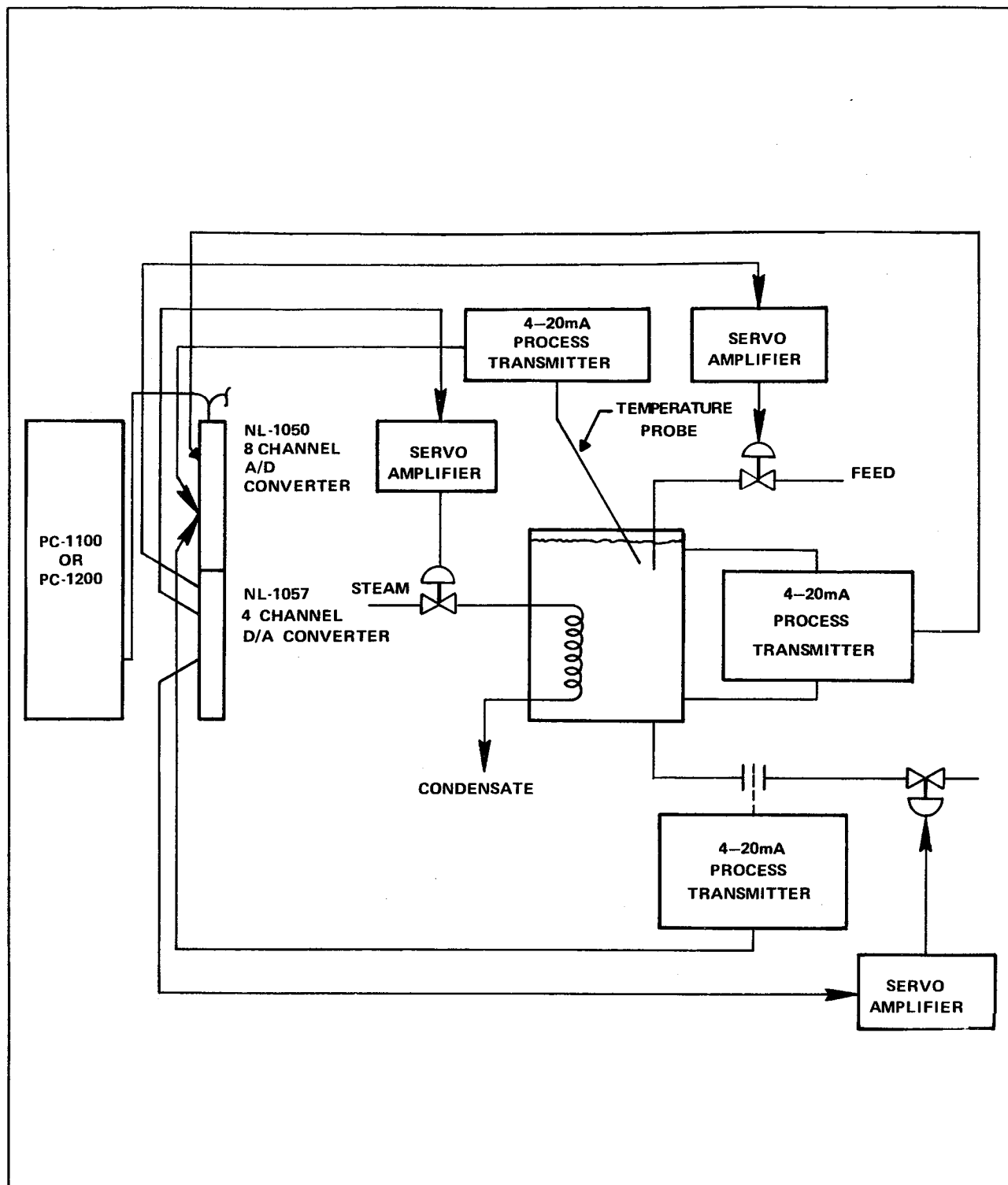


Figure 25. Three-Loop System - Programmable Control Configuration

## LC

- Alarm deadband

$$\sqrt{\frac{0.0255 \text{ units}}{1 \text{ pm}^2}} \times 7 \text{ lpm} = 1.11 \text{ units}$$

~ 1 unit

These values will also be used in the loop table worksheet.

The level control loop has limit alarms. Any time the level attempts to exceed 175 cm, an alarm will activate and other logic will stop the feed. Should the level fall below 50 cm, an alarm will activate and other logic will turn OFF the steam to the system. A deadband of 20 cm is provided.

- Low alarm limit

$$\frac{1.275 \text{ units}}{\text{cm}} \times 50 \text{ cm} = 63.75 \text{ units}$$

~ 63 units

- High alarm limit

$$\frac{1.275 \text{ units}}{\text{cm}} \times 175 \text{ cm} = 223.12 \text{ units}$$

~ 223 units

- Alarm deadband

$$\frac{1.275 \text{ units}}{\text{cm}} \times 20 \text{ cm} = 25.5 \text{ units}$$

~ 25 units

The three loops in this example function independently and are programmed independently, each using a separate worksheet (see Tables 7 through 9). Figure 26 shows the program for this three-loop system.

In the loop tables, the batch unit high limit is programmed at 255 or 9999 (the maximum value in these system), and the deviation and limit alarms are programmed (even if not used). The figures given for the tuning constant are arbitrary and necessitate further adjustment, see Section "Loop Tuning Methods".

TABLE 7. TEMPERATURE CONTROL (LT0010) DATA WORKSHEET

Loop Title:		C	H	I	O	I	O	Register Type & No. (Data If CV)	Comment
	Set Point	*	*	*	*	*	*	IR0008	
	Process Variable	*	*	*	*	*	*	IR0001	
	Output	*	*	*	*	*	*	OR0001	
Loop Coll #:	Loop Table End	*	*	*	*	*	*	HR0036	

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity	Value/Remarks	
			PC-1100	PC-1200
HRXXX-35	HR0001	Reserved	C	
HRXXX-34	HR0002	Derivative Coefficient	C	
HRXXX-33	HR0003	Previous Integral Term	C	
HRXXX-32	HR0004	Integral Coefficient	C	
HRXXX-31	HR0005	Proportional Term ( $\pm 32.767$ )	C	
HRXXX-30	HR0006	Integral Term ( $\pm 32.767$ )	C	
HRXXX-29	HR0007	Derivative Term ( $\pm 32.767$ )	C	
HRXXX-28	HR0008	SP <sub>n</sub> - Set Point This Sample	C	
HRXXX-27	HR0009	PV <sub>n</sub> - Process Variable This Sample	C	
HRXXX-26	HR0010	Time Counter - Elapsed Sample Time	C	
HRXXX-25	HR0011	SP <sub>n-1</sub> - Set Point Previous Sample	C	
HRXXX-24	HR0012	PV <sub>n-1</sub> - Process Variable Previous Sample	C	
HRXXX-23	HR0013	E <sub>n-1</sub> - Error Previous Sample*	C	
HRXXX-22	HR0014	Bias (0 to Maximum Output)	C	
HRXXX-21	HR0015	Functional Gain	C	
HRXXX-20	HR0016	Configuration Input Word (See Below)	U	020FH
HRXXX-19	HR0017	Error Deadband (0-9999)	PC-1200 Only U	
HRXXX-18	HR0018	Derivative Decay Coefficient(0-99.99%)	PC-1200 Only U	
HRXXX-17	HR0019	Integral Sum ( $\pm 32.767$ )	C	
HRXXX-16	HR0020	E <sub>n</sub> - Error This Sample*	C	
HRXXX-15	HR0021	T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	5.00 Minutes
HRXXX-14	HR0022	T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	10.00 Minutes
HRXXX-13	HR0023	T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	100 Tenths of Second
HRXXX-12	HR0024	K <sub>c</sub> - Proportional Gain (0 - 99.99)	U	10.0
HRXXX-11	HR0025	Inner Loop Pointer (Loop Table End)	U	Not used in example
HRXXX-10	HR0026	Outer Loop Pointer (Loop Table End)	U	Not used in example
HRXXX-9	HR0027	Alarm Deadband (0 - Max PV)	U	2 Units
HRXXX-8	HR0028	Batch Unit Preload (0 - Max Output)	U	0000 Default
HRXXX-7	HR0029	Batch Unit Hi Limit (0 - Max Output)	U	255 Units
HRXXX-6	HR0030	Neg. Slew Limit (Max - $\Delta$ Output/Sample)	U	Not used in example
HRXXX-5	HR0031	Pos. Slew Limit (Max + $\Delta$ Output/Sample)	U	Not used in example
HRXXX-4	HR0032	Low Deviation Alarm Limit (0 - Max PV)	U	12 Units
HRXXX-3	HR0033	High Deviation Alarm Limit (0 - Max PV)	U	25 Units
HRXXX-2	HR0034	Low Alarm Limit (0 - Max PV)	U	0000 Default
HRXXX-1	HR0035	High Alarm Limit (0 - Max PV)	U	255 Units
HRXXX	HR0036	Output Status Word	C	

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXX-20)      16 15 14 13      12 11 10 9      8 7 6 5      4 3 2 1      = 020FH

   0 0 0 0      0 0 1 0      0 0 0 0      1 1 1 1

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected	1	9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	0
2	1 = Integral Mode Selected	1	10	1 = Batch Unit Selected	1
3	1 = Derivative Mode Selected	1	11	RESERVED FOR CONTROLLER USE	0
4	1 = Deviation Alarms Selected	1	12	RESERVED FOR FUTURE USE	0
5	1 = Error Deadband Selected	0	13	RESERVED FOR FUTURE USE	0
6	RESERVED FOR FUTURE USE	0	14	PC-1100: Coefficient Lock	0
7	1 = Slew Limiting Selected	0	15	RESERVED FOR FUTURE USE	0
8	1 = Reverse Action Selected 0 = Direct Action Selected	0	16	RESERVED FOR FUTURE USE	0

Table 7

TABLE 8. LEVEL CONTROL (LT0011) DATA WORKSHEET

Loop Title:		C	H	I	O	O	Register Type & No. (Data if CV)	Comment
	Set Point	*	*	*	*	*	HR0109	
	Process Variable	*	*	*	*	*	IR0002	
	Output	*	*	*	*	*	OR0002	
Loop Coll #:	Loop Table End	*	*	*	*	*	HR0072	

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity	Value/Remarks	
			PC-1100	PC-1200
HRXXX-35	HR0037	Reserved	C	
HRXXX-34	HR0038	Derivative Coefficient	C	
HRXXX-33	HR0039	Previous Integral Term	C	
HRXXX-32	HR0040	Integral Coefficient	C	
HRXXX-31	HR0041	Proportional Term ( $\pm 32.767$ )	C	
HRXXX-30	HR0042	Integral Term ( $\pm 32.767$ )	C	
HRXXX-29	HR0043	Derivative Term ( $\pm 32.767$ )	C	
HRXXX-28	HR0044	SP <sub>n</sub> - Set Point This Sample	C	
HRXXX-27	HR0045	PV <sub>n</sub> - Process Variable This Sample	C	
HRXXX-26	HR0046	Time Counter - Elapsed Sample Time	C	
HRXXX-25	HR0047	SP <sub>n-1</sub> - Set Point Previous Sample	C	
HRXXX-24	HR0048	PV <sub>n-1</sub> - Process Variable Previous Sample	C	
HRXXX-23	HR0049	E <sub>n-1</sub> - Error Previous Sample*	C	
HRXXX-22	HR0050	Bias (0 to Maximum Output)	C	
HRXXX-21	HR0051	Functional Gain	C	
HRXXX-20	HR0052	Configuration Input Word (See Below)	U	0207H
HRXXX-19	HR0053	Error Deadband (0-9999)	PC-1200 Only U	
HRXXX-18	HR0054	Derivative Decay Coefficient (0-99.99%)	PC-1200 Only U	
HRXXX-17	HR0055	Integral Sum ( $\pm 32.767$ )	C	
HRXXX-16	HR0056	E <sub>n</sub> - Error This Sample*	C	
HRXXX-15	HR0057	T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	5.00 Minutes
HRXXX-14	HR0058	T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	10.00 Minutes
HRXXX-13	HR0059	T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	1000 Tenths of Second
HRXXX-12	HR0060	K <sub>C</sub> - Proportional Gain (0 - 99.99)	U	10.00
HRXXX-11	HR0061	Inner Loop Pointer (Loop Table End)	U	Not used in example
HRXXX-10	HR0062	Outer Loop Pointer (Loop Table End)	U	Not used in example
HRXXX-9	HR0063	Alarm Deadband (0 - Max PV)	U	25 Units
HRXXX-8	HR0064	Batch Unit Preload (0 - Max Output)	U	0000 Default
HRXXX-7	HR0065	Batch Unit Hi Limit (0 - Max Output)	U	255 Units
HRXXX-6	HR0066	Neg. Slew Limit (Max - $\Delta$ Output/Sample)	U	Not used in example
HRXXX-5	HR0067	Pos. Slew Limit (Max + $\Delta$ Output/Sample)	U	Not used in example
HRXXX-4	HR0068	Low Deviation Alarm Limit (0 - Max PV)	U	Not used in example
HRXXX-3	HR0069	High Deviation Alarm Limit (0 - Max PV)	U	Not used in example
HRXXX-2	HR0070	Low Alarm Limit (0 - Max PV)	U	63 Units
HRXXX-1	HR0071	High Alarm Limit (0 - Max PV)	U	233 Units
HRXXX	HR0072	Output Status Word	C	

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXX-20)

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected	1	9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	0
2	1 = Integral Mode Selected	1	10	1 = Batch Unit Selected	1
3	1 = Derivative Mode Selected	1	11	RESERVED FOR CONTROLLER USE	0
4	1 = Deviation Alarms Selected	0	12	RESERVED FOR FUTURE USE	0
5	1 = Error Deadband Selected	0	13	RESERVED FOR FUTURE USE	0
6	RESERVED FOR FUTURE USE	0	14	PC-1100 Coefficient Lock	0
7	1 = Slew Limiting Selected	0	15	RESERVED FOR FUTURE USE	0
8	1 = Reverse Action Selected 0 = Direct Action Selected	0	16	RESERVED FOR FUTURE USE	0

Table 8

TABLE 9. FLOW CONTROL (LT0012) DATA WORKSHEET

Loop Title:		C	H	I	O	I	O	Register Type & No. (Data If CV)	Comment
	Set Point	*	*	*	*	*	*	HR0114	
	Process Variable	*	*	*	*	*	*	HR0113	
	Output	*	*	*	*	*	*	OR0003	
Loop Coil #:	Loop Table End	*	*	*	*	*	*	HR0108	

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity	Value/Remarks	
			PC-1100	PC-1200
HRXXXX-35	HR0073	Reserved	C	
HRXXXX-34	HR0074	Derivative Coefficient	C	
HRXXXX-33	HR0075	Previous Integral Term	C	
HRXXXX-32	HR0076	Integral Coefficient	C	
HRXXXX-31	HR0077	Proportional Term ( $\pm 32.767$ )	C	
HRXXXX-30	HR0078	Integral Term ( $\pm 32.767$ )	C	
HRXXXX-29	HR0079	Derivative Term ( $\pm 32.767$ )	C	
HRXXXX-28	HR0080	SP <sub>n</sub> - Set Point This Sample	C	
HRXXXX-27	HR0081	PV <sub>n</sub> - Process Variable This Sample	C	
HRXXXX-26	HR0082	Time Counter - Elapsed Sample Time	C	
HRXXXX-25	HR0083	SP <sub>n-1</sub> - Set Point Previous Sample	C	
HRXXXX-24	HR0084	PV <sub>n-1</sub> - Process Variable Previous Sample	C	
HRXXXX-23	HR0085	E <sub>n-1</sub> - Error Previous Sample*	C	
HRXXXX-22	HR0086	Bias (0 to Maximum Output)	C	
HRXXXX-21	HR0087	Functional Gain	C	
HRXXXX-20	HR0088	Configuration Input Word (See Below)	U	020FH
HRXXXX-19	HR0089	Error Deadband (0-9999)	PC-1200 Only U	
HRXXXX-18	HR0090	Derivative Decay Coefficient (0-99.99%)	PC-1200 Only U	
HRXXXX-17	HR0091	Integral Sum ( $\pm 32.767$ )	C	
HRXXXX-16	HR0092	E <sub>n</sub> - Error This Sample*	C	
HRXXXX-15	HR0093	T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	5.00 Minutes
HRXXXX-14	HR0094	T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	10.00 Minutes
HRXXXX-13	HR0095	T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	100 Tenths of Second
HRXXXX-12	HR0096	K <sub>c</sub> - Proportional Gain (0 - 99.99)	U	10.00
HRXXXX-11	HR0097	Inner Loop Pointer (Loop Table End)	U	Not used in example
HRXXXX-10	HR0098	Outer Loop Pointer (Loop Table End)	U	Not used in example
HRXXXX-9	HR0099	Alarm Deadband (0 - Max PV)	U	2 Units
HRXXXX-8	HR0100	Batch Unit Preload (0 - Max Output)	U	0000 Default
HRXXXX-7	HR0101	Batch Unit Hi Limit (0 - Max Output)	U	255 Units
HRXXXX-6	HR0102	Neg. Slew Limit (Max - ΔOutput/Sample)	U	Not used in example
HRXXXX-5	HR0103	Pos. Slew Limit (Max + ΔOutput/Sample)	U	Not used in example
HRXXXX-4	HR0104	Low Deviation Alarm Limit (0 - Max PV)	U	3 Units
HRXXXX-3	HR0105	High Deviation Alarm Limit (0 - Max PV)	U	1 Unit
HRXXXX-2	HR0106	Low Alarm Limit (0 - Max PV)	U	0000 Default
HRXXXX-1	HR0107	High Alarm Limit (0 - Max PV)	U	255 Units
HRXXXX	HR0108	Output Status Word	C	

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXXX-20)      16 15 14 13      12 11 10 9      8 7 6 5      4 3 2 1      = 020FH

0 0 0 0      0 0 1 0      0 0 0 0      1 1 1 1

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected	1	9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	0
2	1 = Integral Mode Selected	1	10	1 = Batch Unit Selected	1
3	1 = Derivative Mode Selected	1	11	RESERVED FOR CONTROLLER USE	0
4	1 = Deviation Alarms Selected	1	12	RESERVED FOR FUTURE USE	0
5	1 = Error Deadband Selected	0	13	RESERVED FOR FUTURE USE	0
6	RESERVED FOR FUTURE USE	0	14	PC-1100: Coefficient Lock	0
7	1 = Slew Limiting Selected	0	15	RESERVED FOR FUTURE USE	0
8	1 = Reverse Action Selected 0 = Direct Action Selected	0	16	RESERVED FOR FUTURE USE	0

Table 9

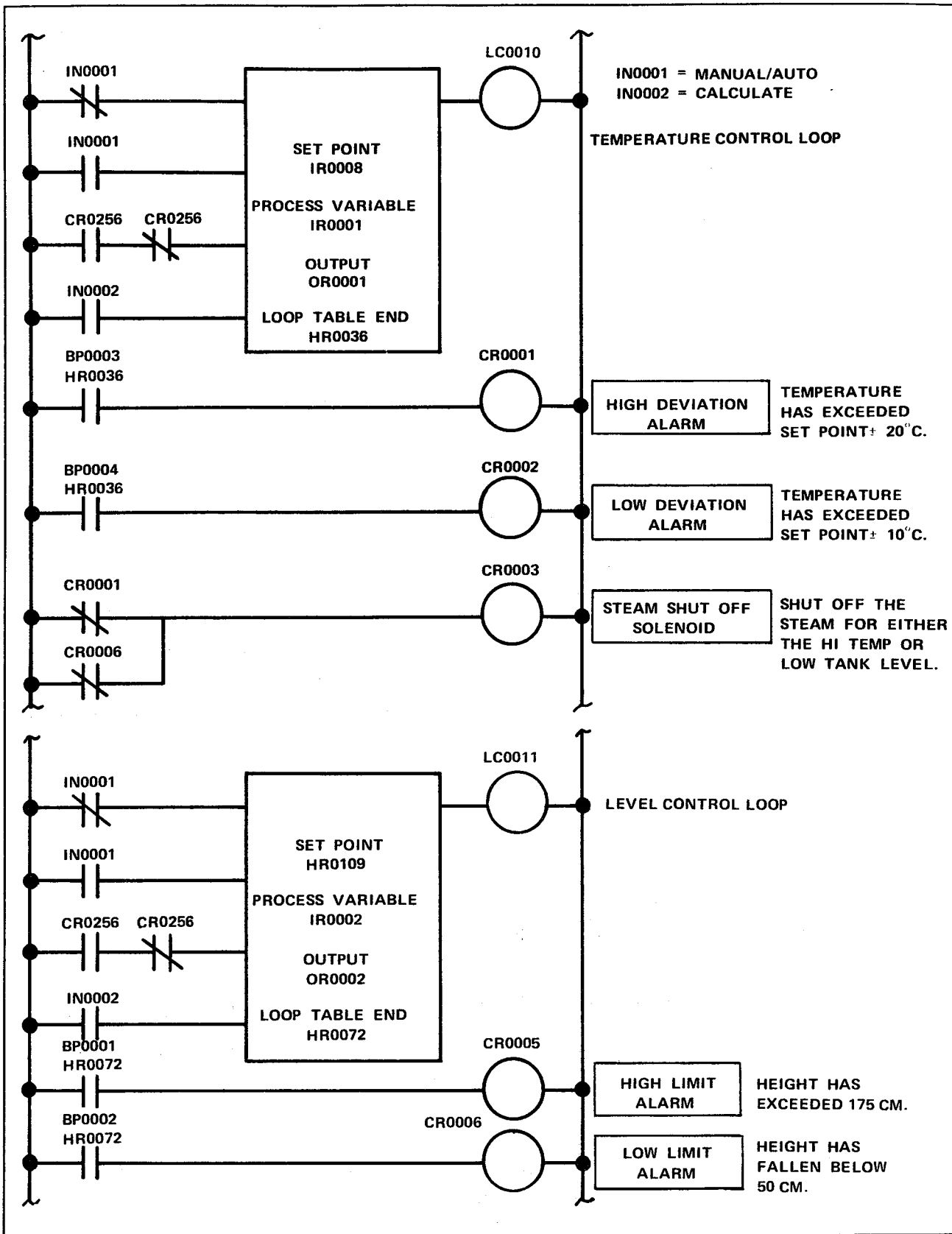


Figure 26a. Three-Loop Program

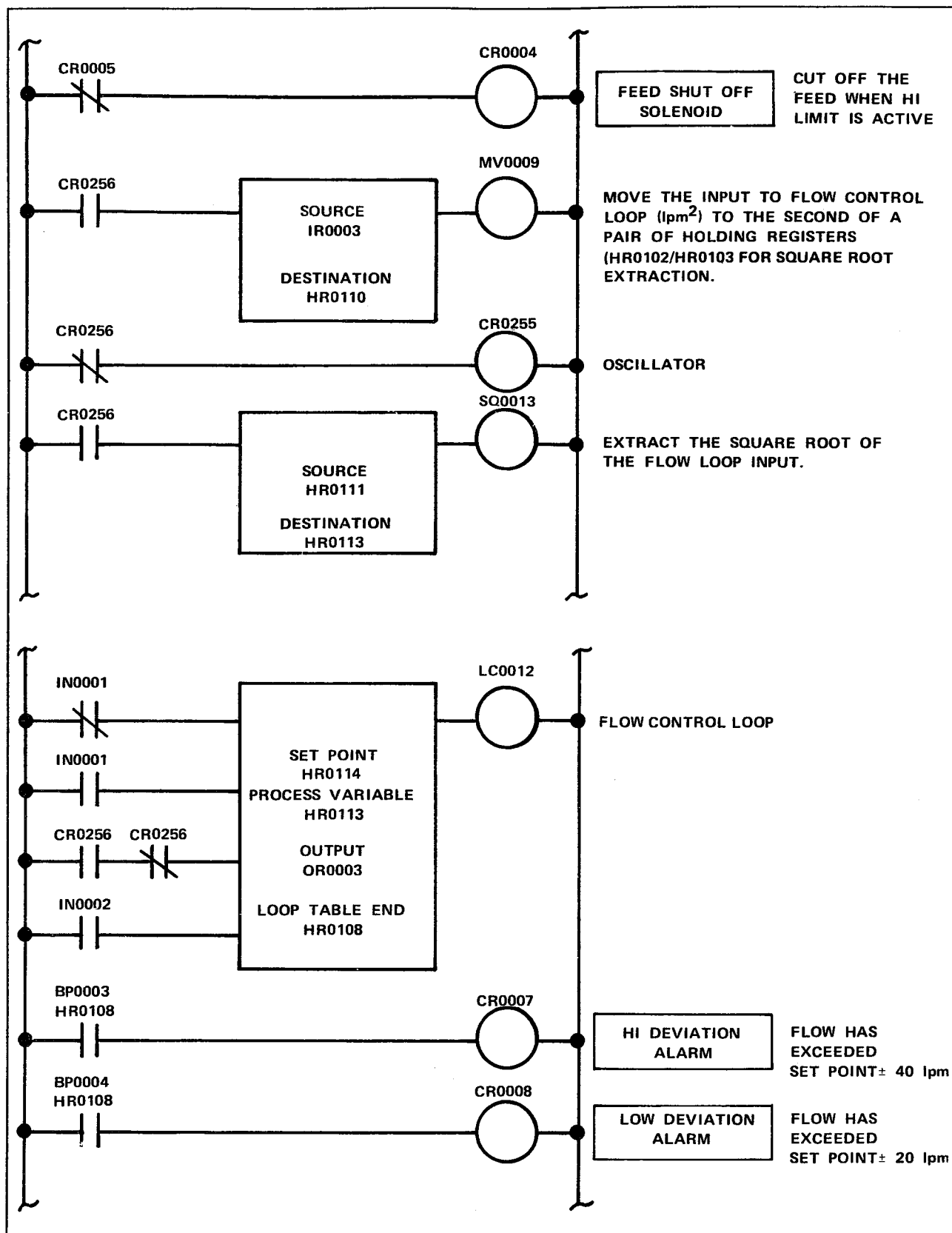


Figure 26b. Three-Loop Program (Cont'd)



Cascade control is used when a process variable can be adversely affected by another process variable. Cascading loops allows a main loop to control one or more minor loops. In the previous example, steam was used to heat a product and a single temperature control loop was used to control temperature. In that system, a variation in the flow of steam would have been corrected, but only after temperature had actually changed. An improvement in performance of the temperature control could have been brought about by using Cascade control, as shown in Figure 27. In the inner loop, flow is directly controlled to a set point established by the temperature control loop output. Fluctuations in flow are dealt with directly - before they cause a change in temperature. This, in effect, removes steam flow as a variable of any consequence in this process and provides very close control of temperature.

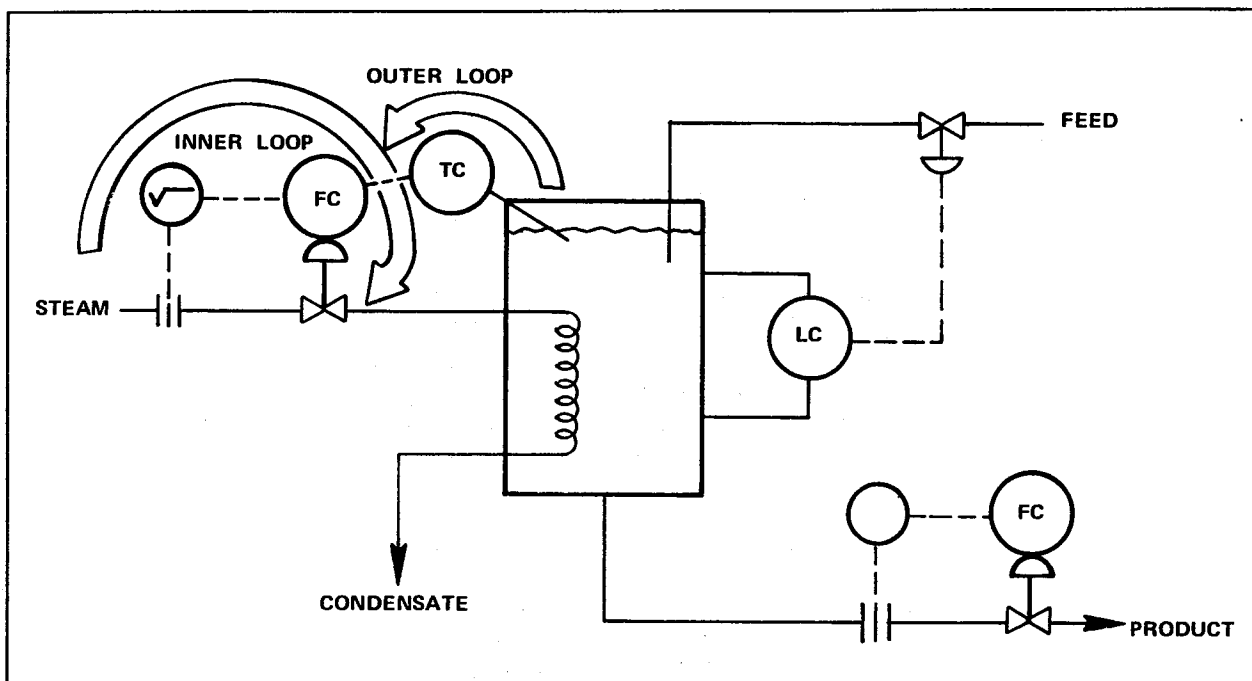


Figure 27. Cascade Control

The same control program in Figure 26 can be used for the level control loop and the product flow control loop; however, the temperature control loop must be modified as follows (see Figures 28 and 29):

1. The output will be placed in a holding register location for use in programming the inner loop set point.

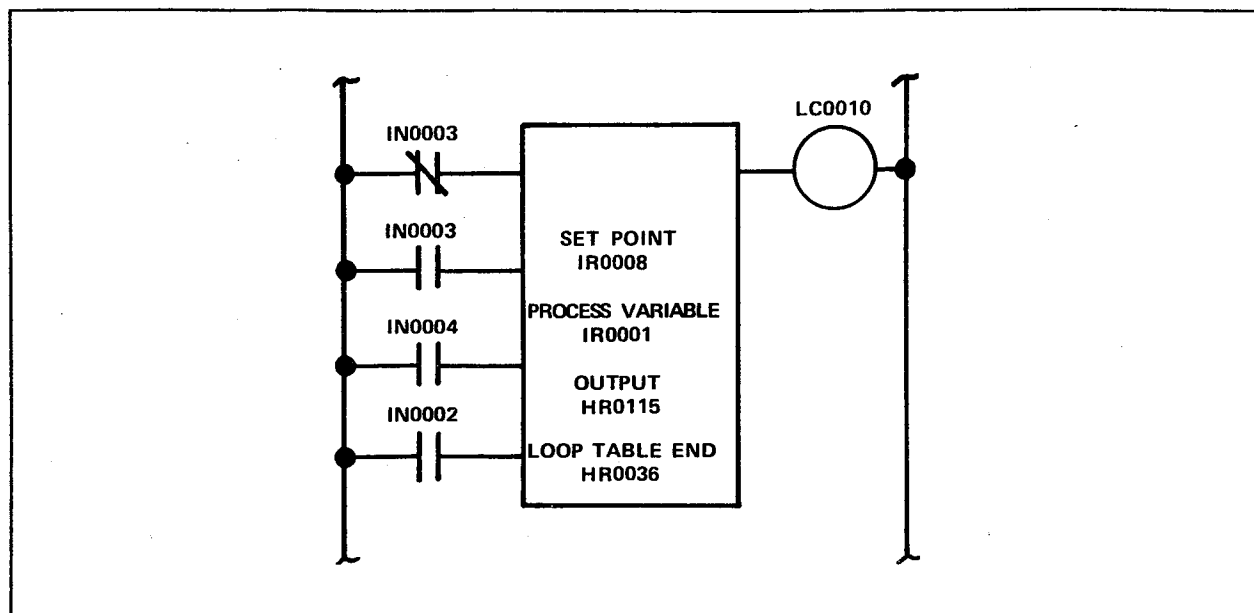


Figure 28. Cascade Program: LC

2. The loop table must be modified as follows:

HR0021 = (HR) 0161 inner loop pointer for PC-1100

HR0025 = (HR) 0161 inner loop pointer for PC-1200

The inner loop, which is a flow control loop whose process variable input is the square of flow, will require that the square root of the input be found. (See Figure 29.)

Replacing LC0010 in Figure 26 with the modified LC0010 of Figure 18 and LC0013, V0015 and LT0016 of Figure 29 results in a cascade control system.

The inner and outer loop pointers (table ends) must be specified in the tables for each of the loops. (See Tables 10 and 11.) When the system is configured in this manner, temperature is more closely controlled than by a simple temperature control loop.

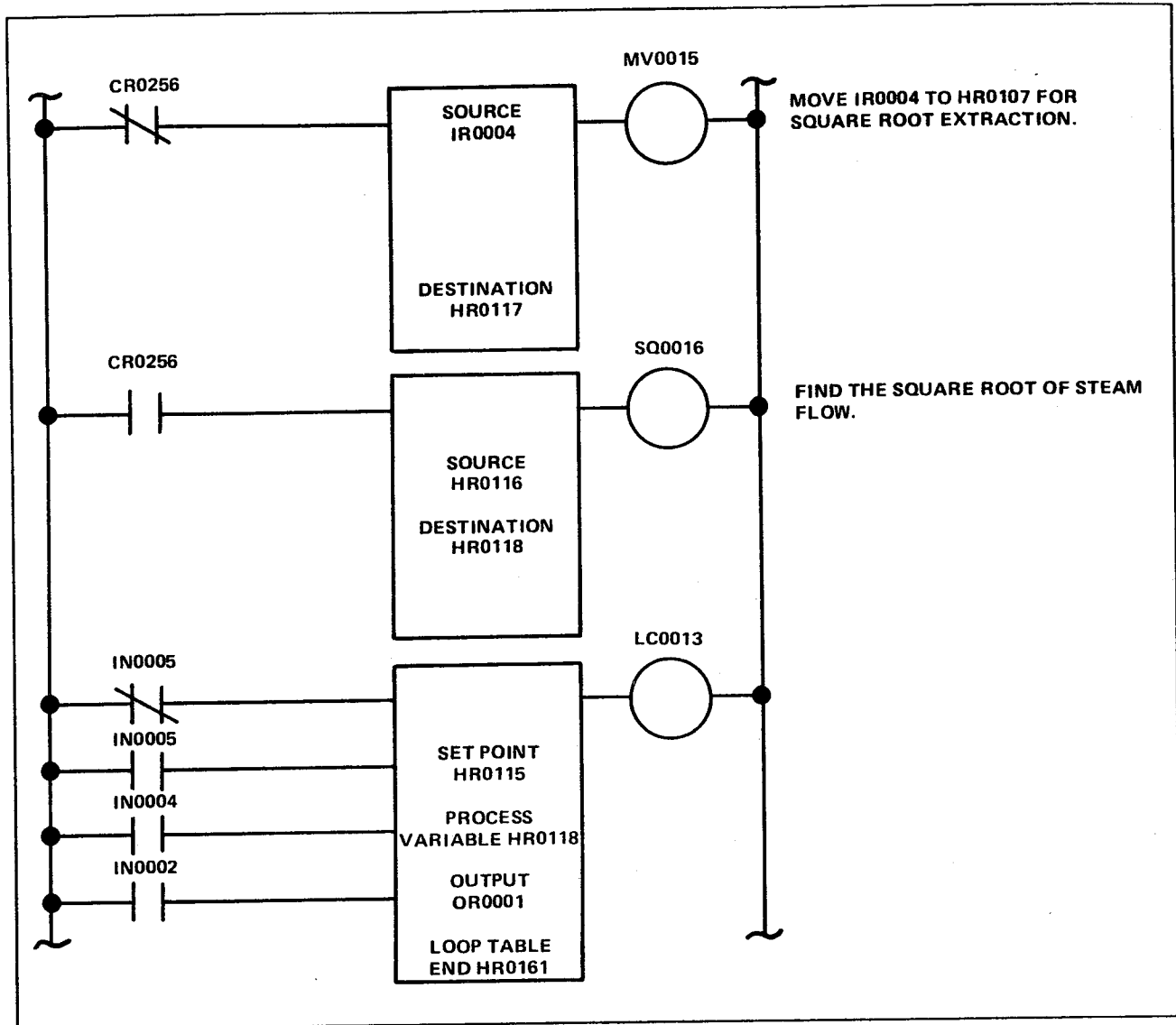


Figure 29. Cascade Program: Square Root

Cascaded loops may be tuned using any of the methods described in Section "Loop Tuning Methods". However, the following procedure must be followed.

1. Set both loops to Manual
2. Load the desired outer loop set point.
3. Adjust the inner loop output until the set point equals the process variable on the output loop.
4. Adjust the set point of the inner loop until the set point equals the process variable on the inner loop.

TABLE 10. TEMPERATURE CONTROL (LT0010) - CASCADE OUTER LOOP DATA WORKSHEET

Loop Title:		C	H	I	O	I	O	Register Type & No. (Data if CV)	Comment	
	Set Point	*	*	*	*	*	*			IR0008
	Process Variable		*	*	*	*	*			IR0001
	Output		*	*	*	*	*			HR0115
Loop Coil #:	Loop Table End	*						HR0036		

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity		Value/Remarks	
				PC-1100	PC-1200
HRXXX-35	HR0001	Reserved	C		
HRXXX-34	HR0002	Derivative Coefficient	C		
HRXXX-33	HR0003	Previous Integral Term	C		
HRXXX-32	HR0004	Integral Coefficient	C		
HRXXX-31	HR0005	Proportional Term (± 32.767)	C		
HRXXX-30	HR0006	Integral Term (± 32.767)	C		
HRXXX-29	HR0007	Derivative Term (± 32.767)	C		
HRXXX-28	HR0008	SP <sub>n</sub> - Set Point This Sample	C		
HRXXX-27	HR0009	PV <sub>n</sub> - Process Variable This Sample	C		
HRXXX-26	HR0010	Time Counter - Elapsed Sample Time	C		
HRXXX-25	HR0011	SP <sub>n-1</sub> - Set Point Previous Sample	C		
HRXXX-24	HR0012	PV <sub>n-1</sub> - Process Variable Previous Sample	C		
HRXXX-23	HR0013	E <sub>n-1</sub> - Error Previous Sample*	C		
HRXXX-22	HR0014	Bias (0 to Maximum Output)	C		
HRXXX-21	HR0015	Functional Gain	C	020FH	
HRXXX-20	HR0016	Configuration Input Word (See Below)	U		
HRXXX-19	HR0017	Error Deadband (0-9999)	U	PC-1200 Only	
HRXXX-18	HR0018	Derivative Decay Coefficient (0-99.99%)	U	PC-1200 Only	
HRXXX-17	HR0019	Integral Sum (± 32.767)	C		
HRXXX-16	HR0020	E <sub>n</sub> - Error This Sample*	C		
HRXXX-15	HR0021	T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	5.00 Minutes	
HRXXX-14	HR0022	T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	10.00 Minutes	
HRXXX-13	HR0023	T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	100 tenths of second	
HRXXX-12	HR0024	K <sub>c</sub> - Proportional Gain (0 - 99.99)	U	10.00	
HRXXX-11	HR0025	Inner Loop Pointer (Loop Table End)	U	HR0161	
HRXXX-10	HR0026	Outer Loop Pointer (Loop Table End)	U	Not used in example	
HRXXX-9	HR0027	Alarm Deadband (0 - Max PV)	U	2 Units	
HRXXX-8	HR0028	Batch Unit Preload (0 - Max Output)	U	0000 Default	
HRXXX-7	HR0029	Batch Unit Hi Limit (0 - Max Output)	U	255 Units	
HRXXX-6	HR0030	Neg. Slew Limit (Max - ΔOutput/Sample)	U	Not used in example	
HRXXX-5	HR0031	Pos. Slew Limit (Max + ΔOutput/Sample)	U	Not used in example	
HRXXX-4	HR0032	Low Deviation Alarm Limit (0 - Max PV)	U	12 Units	
HRXXX-3	HR0033	High Deviation Alarm Limit (0 - Max PV)	U	25 Units	
HRXXX-2	HR0034	Low Alarm Limit (0 - Max PV)	U	0000 Default	
HRXXX-1	HR0035	High Alarm Limit (0 - Max PV)	U	255 Units	
HRXXX	HR0036	Output Status Word	C		

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXX-20)      16 15 14 13      12 11 10 9      8 7 6 5      4 3 2 1  
 0 0 0 0      0 0 1 0      0 0 0 0      1 1 1 1      = 020FH

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected	1	9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	0
2	1 = Integral Mode Selected	1	10	1 = Batch Unit Selected	1
3	1 = Derivative Mode Selected	1	11	RESERVED FOR CONTROLLER USE	0
4	1 = Deviation Alarms Selected	1	12	RESERVED FOR FUTURE USE	0
5	1 = Error Deadband Selected	0	13	RESERVED FOR FUTURE USE	0
6	RESERVED FOR FUTURE USE	0	14	PC-1100: Coefficient Lock	0
7	1 = Slew Limiting Selected	0	15	RESERVED FOR FUTURE USE	0
8	1 = Reverse Action Selected 0 = Direct Action Selected	0	16	RESERVED FOR FUTURE USE	0

Table 10

TABLE 11. FLOW CONTROL (LT0013) - CASCADE INNER LOOP DATA WORKSHEET

Loop Title:		C	H	I	O	I	O	Register Type & No. (Data if CV)	Comment
	Set Point	*	*	*	*	*	*	HR0115	
	Process Variable		*	*	*	*	*	HR0118	
	Output		*	*	*	*	*	OR0001	
Loop Coll #:	Loop Table End	*	*	*	*	*	*	HR0161	

Loop Table Register Positions	Loop Table Actual HR Assignment	Quantity	Value/Remarks	
			PC-1100	PC-1200
HRXXXX-35	HR0126	Reserved	C	
HRXXXX-34	HR0127	Derivative Coefficient	C	
HRXXXX-33	HR0128	Previous Integral Term	C	
HRXXXX-32	HR0129	Integral Coefficient	C	
HRXXXX-31	HR0130	Proportional Term ( $\pm 32.767$ )	C	
HRXXXX-30	HR0131	Integral Term ( $\pm 32.767$ )	C	
HRXXXX-29	HR0132	Derivative Term ( $\pm 32.767$ )	C	
HRXXXX-28	HR0133	SP <sub>n</sub> - Set Point This Sample	C	
HRXXXX-27	HR0134	PV <sub>n</sub> - Process Variable This Sample	C	
HRXXXX-26	HR0135	Time Counter - Elapsed Sample Time	C	
HRXXXX-25	HR0136	SP <sub>n-1</sub> - Set Point Previous Sample	C	
HRXXXX-24	HR0137	PV <sub>n-1</sub> - Process Variable Previous Sample	C	
HRXXXX-23	HR0138	E <sub>n-1</sub> - Error Previous Sample*	C	
HRXXXX-22	HR0139	Bias (0 to Maximum Output)	C	
HRXXXX-21	HR0140	Functional Gain	C	
HRXXXX-20	HR0141	Configuration Input Word (See Below)	U	0207H
HRXXXX-19	HR0142	Error Deadband (0-9999)	U	PC-1200 Only
HRXXXX-18	HR0143	Derivative Decay Coefficient (0-99.99%)	U	PC-1200 Only
HRXXXX-17	HR0144	Integral Sum ( $\pm 32.767$ )	C	
HRXXXX-16	HR0145	E <sub>n</sub> - Error This Sample*	C	
HRXXXX-15	HR0146	T <sub>d</sub> - Derivative Time (0 - 327.67 Min.)	U	5.00 Minutes
HRXXXX-14	HR0147	T <sub>i</sub> - Integral Time (0 - 327.67 Min.)	U	10.00 Minutes
HRXXXX-13	HR0148	T <sub>s</sub> - Sample Time (0 - 3276.7 Sec.)	U	100 tenths of second
HRXXXX-12	HR0149	K <sub>c</sub> - Proportional Gain (0 - 99.99)	U	5.00
HRXXXX-11	HR0150	Inner Loop Pointer (Loop Table End)	U	Not used in example
HRXXXX-10	HR0151	Outer Loop Pointer (Loop Table End)	U	HR0032
HRXXXX-9	HR0152	Alarm Deadband (0 - Max PV)	U	Not used in example
HRXXXX-8	HR0153	Batch Unit Preload (0 - Max Output)	U	0000 Default
HRXXXX-7	HR0154	Batch Unit Hi Limit (0 - Max Output)	U	255 Units
HRXXXX-6	HR0155	Neg. Slew Limit (Max - $\Delta$ Output/Sample)	U	Not used in example
HRXXXX-5	HR0156	Pos. Slew Limit (Max + $\Delta$ Output/Sample)	U	Not used in example
HRXXXX-4	HR0157	Low Deviation Alarm Limit (0 - Max PV)	U	0000
HRXXXX-3	HR0158	High Deviation Alarm Limit (0 - Max PV)	U	255 Units
HRXXXX-2	HR0159	Low Alarm Limit (0 - Max PV)	U	0000 Default
HRXXXX-1	HR0160	High Alarm Limit (0 - Max PV)	U	255 Units
HRXXXX	HR0161	Output Status Word	C	

C = Calculated by Processor      U = User-Entered      \* = 2's Complement Signed Value

Configuration Input Word (HRXXXX-20)      16 15 14 13      12 11 10 9      8 7 6 5      4 3 2 1      = 0207H

   0 0 0 0      0 0 1 0      0 0 0 0      0 1 1 1

Bit Number	Definition	Status	Bit Number	Definition	Status
1	1 = Proportional Mode Selected	1	9	1 = Derivative on PV Selected 0 = Derivative on Error Selected	0
2	1 = Integral Mode Selected	1	10	1 = Batch Unit Selected	1
3	1 = Derivative Mode Selected	1	11	RESERVED FOR CONTROLLER USE	0
4	1 = Deviation Alarms Selected	0	12	RESERVED FOR FUTURE USE	0
5	1 = Error Deadband Selected	0	13	RESERVED FOR FUTURE USE	0
6	RESERVED FOR FUTURE USE	0	14	PC-1100 Coefficient Lock	0
7	1 = Slew Limiting Selected	0	15	RESERVED FOR FUTURE USE	0
8	1 = Reverse Action Selected 0 = Direct Action Selected	0	16	RESERVED FOR FUTURE USE	0

Table 11

5. Place the inner loop in Auto. This sets the bias of the inner loop equal to the output of the inner loop and zeroes the integral term and error.
6. Put the inner loop in Cascade. This transfers the output of the outer loop to the set point of the inner loop.
7. Put the outer loop in Auto. This sets the bias of the outer loop equal to the output of the outer loop and zeroes the integral term and error.

## LOOP TUNING METHODS

The LC function operates by using a discrete form of the continuous PID control equation.

- Continuous Equation

$$M(t) = K_c \left[ e(t) + \frac{1}{T_i} \int_0^t e(t) dt + T_d \frac{de(t)}{dt} \right] + M(0)$$

- Discrete Equation

$$M_n = K_c \left[ E_n + \frac{T_s}{T_i} \sum_{j=0}^{j=n} \frac{E_j + E_{j-1}}{2} + \frac{T_d}{T_s} (E_n - E_{n-1}) + (D_{n-1} \times \text{DDc}^*) \right] + M_0$$

where:

$M_n$  = LC output this nth iteration

$M_0$  = Bias (LC output at start)

$K_c$  = Overall proportional gain

$T_i$  = Integral time (I gain)

$T_d$  = Derivative time (D gain)

$E_n$  = Error this nth iteration

\* DDc = Derivative Decay Coefficient (PC-1200 only)

n = nth iteration

These parameters must be established by the user:  $K_c$ ,  $T_i$ ,  $T_d$ ,  $T_s$  and DDc when using the PC-1200. The development of these quantities is called "tuning the loop". To date, there are no foolproof methods of establishing these parameters at the loop design state. Many control loops are, in fact, started and brought to optimum operating conditions, and then tuned.

### Individual PID Term and Gain Description

The proportional term is equal to :

$$P_n = K_c \times E_n$$

$K_c$  is a distributed element through all three terms P,I, and D. It is defined in register HRXXXX-12 of the Loop register Table. This register can be programmed to an actual integer decimal value of 1 to 9999 hundredths of unity gain. In turn, this entry has an assumed decimal point if the units are changed to unity. For example, a decimal integer entry of 100 is equivalent to 1.00 gain.  $K_c$  is a dimensionless gain term.

The integral term is equal to:

$$I = \frac{K_c \times T_s}{T_i} \sum_{j=0}^{j=n} \frac{E_j + E_{j-1}}{2}$$

$T_s$  is used here and in the Derivative term. It is defined in HRXXXX-13 of the Loop Register Table. This register can be programmed to an actual integer decimal value of 1 to 32767 tenths of a second (a meaningful range is typically 1 - 600 ). In turn, this entry has an assumed decimal point if the units are changed to seconds. For example, a decimal integer entry of 27 tenths of a second is equivalent to 2.7 seconds.

$T_i$  is defined in HRXXXX-14 of the Loop Table. This entry can be programmed to an actual integer decimal value of 0 to 32767 hundredths of a minute. In turn, this entry has an assumed decimal point if the units are changed to minutes. For example, a decimal integer entry of 150 hundredths of a minute is equivalent to 1.50 minutes

$T_i$  is not in units of "Resets per Minute".  $T_i$  is in the denominator of three terms to the left of the summation which together are equivalent. The effect of increasing  $T_i$  (with  $T_s$  and  $K_c$  constant) decreases the magnitude of the overall I gain term.

The Derivative Term in the PC-1100 is equal to:

$$D = \frac{K_c \times T_d}{T_s} E_n - E_{n-1}$$

The Derivative Term in the PC-1200 is equal to:

$$D = \frac{K_c \times T_d}{T_s} (e_n - e_{n-1}) + (D_{n-1} \times DDc)$$

$T_d$  is defined in HRXXXX-15 of the Loop Register Table. This register can be programmed to an actual integer decimal value of 0 to 32767 hundredths of a minute. In turn, if the units are changed to minutes this entry will have an assumed decimal place. For example, an entry of 10 defines a derivative time to be 0.10 minute.

Unlike  $T_i$ , increasing  $T_d$  (with  $K_c$  and  $T_s$  constant) increases the weight of the D term.

The PC-1100 form is a simple backwards difference, while the PC-1200 has an exponential decay term to give a more accurate representation of the continuous  $de(t)/dt$  portion of the equation. The "Derivative Decay Coefficient" (DDc) is a percentage gain term which forms the decaying exponential tail. The range for DDc is 0 to 9999 hundredths of a percent and therefore has an assumed decimal point if we change the units to percent. For example, an entry of 9000 is equivalent to 90.00 %.

The effect on the Derivative term to a step Set Point change in Figure 30.

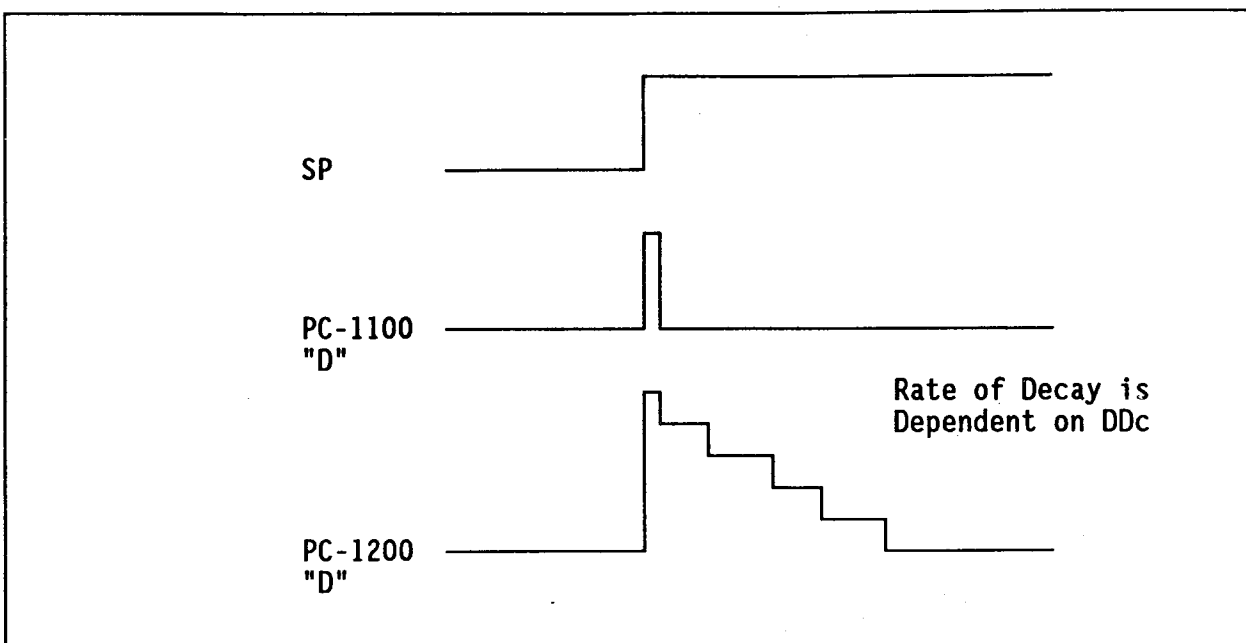


Figure 30. Derivative Term Effects on Step Set Point



There are two basic approaches to loop tuning:

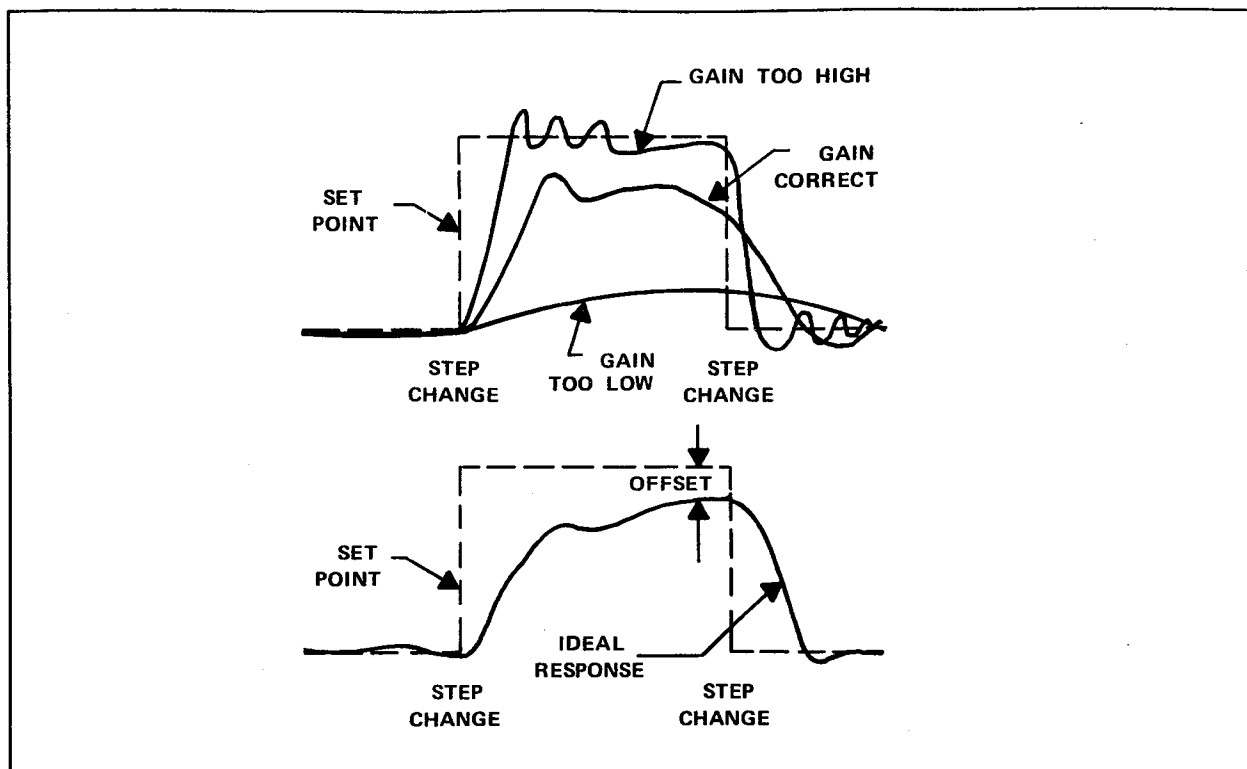
1. Establish and enter the appropriate values. Observe the performance; then, modify the settings.
2. Use a tuning technique involving a process test to obtain data from which settings can be determined. Most often, the first technique is used, but this technique may be more valuable in critical processes.

When using the first method, a systematic approach will yield faster, better results:

1. Delete the Integral (I) and Derivative (D) modes by setting the I and D bits in the Configuration Input Word to zero; this leaves only the Proportional mode. Put the controller in Manual; adjust the output until the desired process variable is achieved. (Select a value within the operating range and equal to the desired steady state value for start-up.) Switch to Auto. (The controller stores the output in a bias register, and zeroes the P, I and D terms, along with e.) Adjust the proportional gain,  $K_c$ , for desired behavior (start with 1.00 through 5.00; adjust for a better response).
2. If the derivative is to be used, adjust  $T_d$  and, if necessary, readjust  $K_c$ . Then activate the Derivative mode by setting the D bit in the Configuration Input Word to 1.
3. If the integral is to be used, adjust  $T_i$  for the desired operation. (It should not be necessary to adjust either  $K_c$  or  $T_d$ .) Then, activate the Integral Mode by setting the I bit in the Configuration Proportional Gain ( $K_c$ ) Input Word to 1.

With the integral and derivative control removed, only  $K_c$  must be adjusted. The following procedure may be helpful:

1. Ensure that the I and D bits of the Configuration/Input Word are set to zero.
2. Allow the process to stabilize (line out) at the desired operation condition.
3. Place the loop into Auto.
4. Quickly move up the set point (i.e., introduce a step change).
5. Observe the system response and adjust for proper response as shown in Figure 31.



**Figure 31. Effects of Proportional Gain**

6. Return the set point to the original setting and observe the response. Figure 31 can be used.
7. Continue the step changes until the loop is adjusted as desired.

**Note**

The higher the gain, the smaller the offset; the lower the gain, the larger the offset.

**Note**

Activating Derivative (or Integral) Control with  $T_d$  (or  $T_i$ ) (equal to zero can cause a user software fault.

**Integral (Reset) Time ( $T_i$ )**

If the controller is to operate without offset, some integral action can be introduced. The ability of a PI loop to operate without offset makes it the most popular form of the control. To adjust  $T_i$ , the following steps can be followed (see Figure 32):

1. Adjust the proportional gain as discussed previously.
2. Activate the Integral control mode by setting the appropriate bit in the Configuration Input Word to 1.
3. Adjust  $T_i$  until the desired operation is achieved.

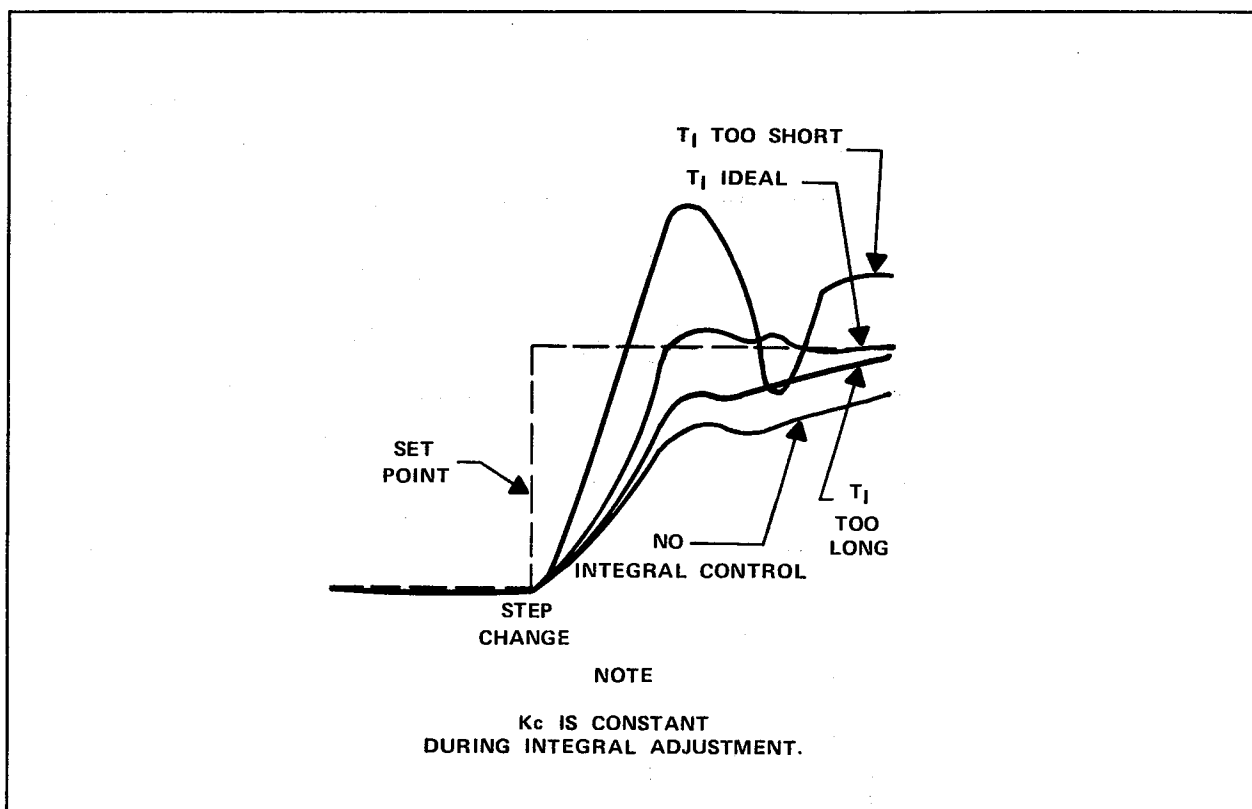


Figure 32. PI Response

### Derivative (Rate) Control

Almost all process loops can benefit from derivative (rate) control, but it is most difficult to properly adjust. There are at least two cases where derivative control is not beneficial:

1. When the process variable is noisy. (Noise induces a false response.)
2. When the process has a large deadtime/transport lag (over the response of the controller).

Other cases can be found that are not benefited by derivative (rate) control; these are beyond the scope of this document. The following procedure is suggested for derivative adjustment.

1. Adjust  $K_c$  with the Integral (Reset) mode deactivated.
2. Activate and adjust  $T_d$ . This may result in the need to increase  $K_c$ .
3. Activate and adjust  $T_i$ .

#### Note

Too much derivative control will cause oscillation.

The effects of derivative action are shown in Figure 33.

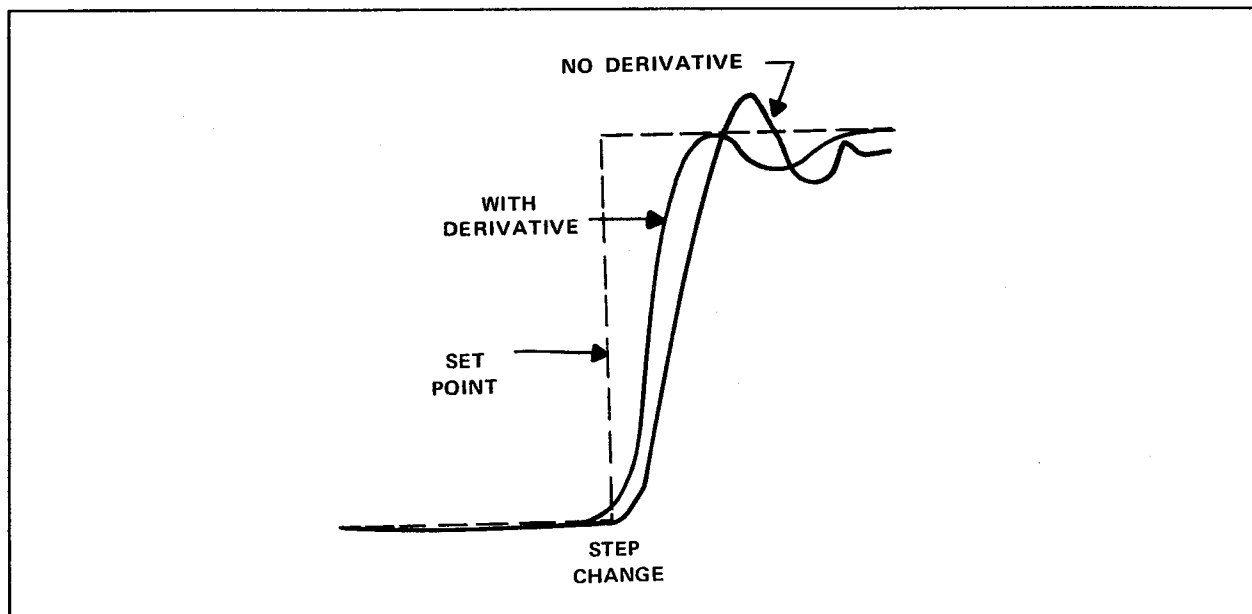


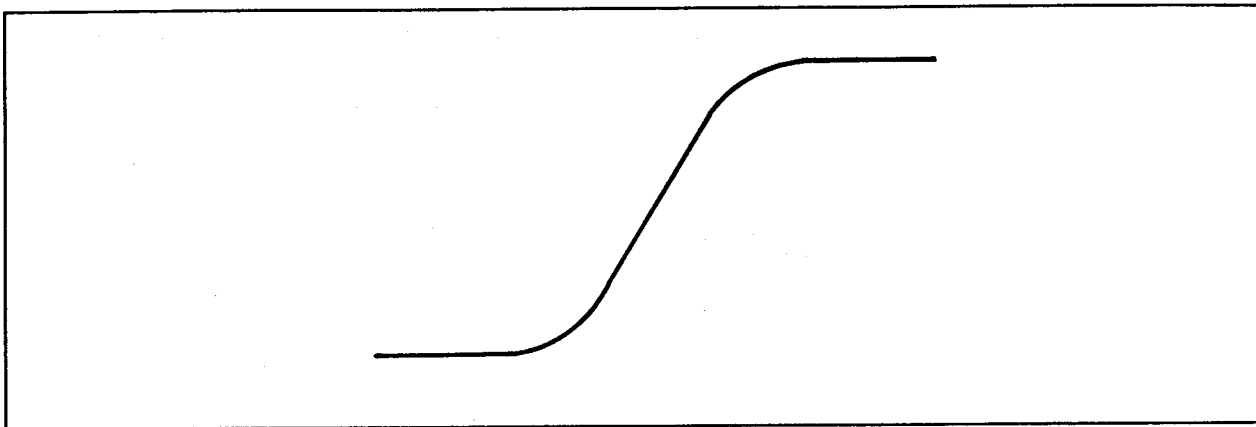
Figure 33. Effects of Derivative Action

## TUNING TECHNIQUES

The following tuning technique attempts to avoid the many step changes involved in Manual tuning. A process test is used that provides data from which initial settings can be calculated. The test procedure is:

1. Line out the process.
2. Place the controller in Manual.
3. Change the output (up or down) by approximately 10 percent of its range, and level it at the new setting. Record the old and new settings.
4. Record the response of the process variable.

The result of this test is a process reaction curve. (See Figure 34). A good quality recorder is required; effort should be made to prevent outside disturbances to the process during testing.



**Figure 34. Typical Process Reaction Curve**

The Ziegler-Nichols tuning procedure requires that two parameters be developed from this curve:

- Lr (Reaction lag)
- Rr (Reaction rate)

These parameters are determined graphically, as shown in Figure 35.

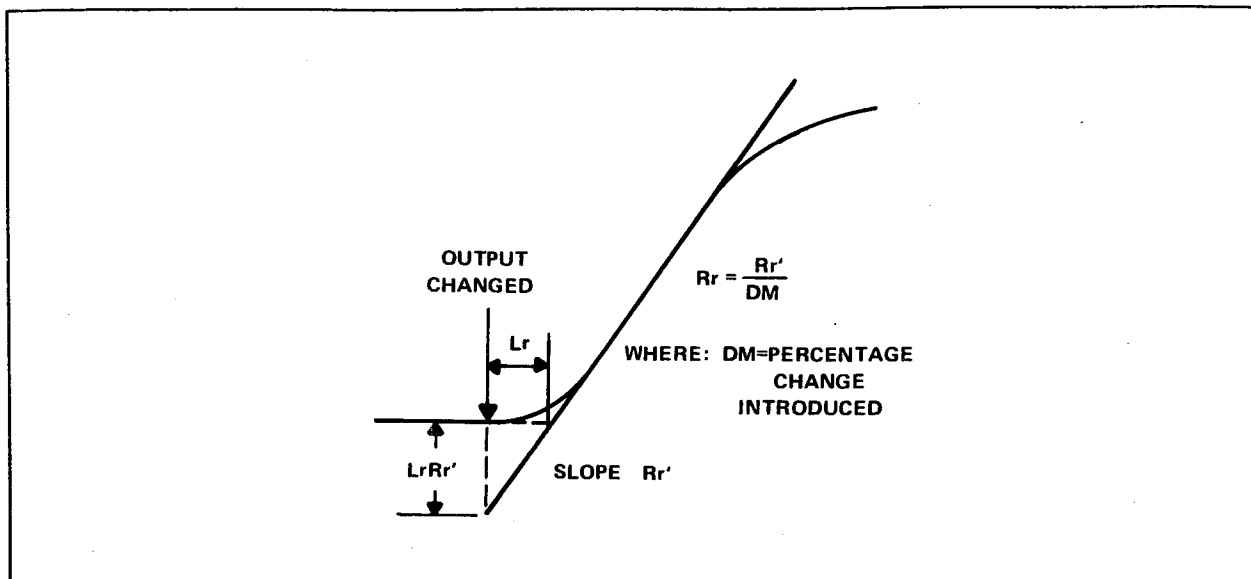


Figure 35. Calculations Using Process Reaction Curve

The following can be calculated by using the quantities determined graphically from Figure 34.

- Proportional gain (for proportional controllers)

$$K_C = \frac{1}{LrRr}$$

- PI (for PI controllers)

$$K_C = \frac{0.9}{LrRr}$$

- PID (for PID controllers)

$$K_C = \frac{1.2}{LrRr}$$

$$T_i = 2.0 Lr$$

$$T_d = 0.5 Lr$$

Although this method is somewhat crude, it yields usable results. Using such a method provides "ball park" initial settings and may reduce the number of further adjustments that must be made.

Loop tuning by any method requires an intimate knowledge of the process under control. The resultant system operation is determined by the process under control, the control system used, and the person tuning the loop.

The guidelines included in this section provide a starting point from which the user can develop the tuning method best suited to system requirements.