# This Errata Sheet contains corrections or changes made after the publication of this manual. 

| Product Family: | DL305 | Date: | 12.04 .2023 |
| :--- | :--- | :--- | :--- |
| Manual Number | D3-USER-M |  |  |
| Revision and Date | 1st Edition, Rev. D; January 2010 |  |  |

### 12.04.2023

F3-04DA-1 module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.
11.16.2023

F3-08TAS-1 module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.
06.21.2023

D3-08TD2, D3-08ND2, D3-16TA-2, and D3-08NA-2 modules, as well as D3-340-EE EPROM chips have been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.
06.01.2023

D3-08TD1 module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.
08.18.2022

D3-232-DCU communications module has been discontinued with no replacement. F3-16TA-2 I/O module has been discontinued. Consider BRX, Productivity, or CLICK PLC systems as upgrades.
06.24.2022

D3-16TD2 Output module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.
05.25.2022

D3-16TD1-1 Output module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

Errata Sheet

# This Errata Sheet contains corrections or changes made after the publication of this manual. 

### 04.08.2022

D3-08NA-1 module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 03.04.2022

03/2022 - D3-08TR module has been discontinued with no replacement. D3-05BDC \& D3-10BDC have been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 01.27 .2022

01/2022 - D3-08TA-1 module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 12.2021

12/2021 - D3-350 CPU module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 11.2021

11/2021 - D3-340 CPU module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 10.21.2021

10/2021 - D3-16NA module has been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 10.14.2021

10/2021-10 slot base D3-10B-1, D3-EXCBL expansion I/O cable, and Handheld Programmer cable D3-HPCBL have been discontinued with no replacement. Consider BRX, Productivity, or CLICK PLC systems as upgrades.

### 09.20.2021

D3-08TA-2 and D3-05B-1 are discontinued as of 09/2021. Please consider Productivity, BRX, or CLICK PLC systems as a replacement.

### 08.26.2021

D3-330 is discontinued as of 08/2021. Please consider the D3-340 or D3-350 CPUs as a replacement.

### 07.28.2021

D3-HP is discontinued as of 03/2021. D3-HPP was discontinued 01/2018. Please consider the Productivity, BRX, or CLICK Series of PLC systems as a replacement.

### 07.16.2021

## Changes to Chapter 5 \& 6.

D3-16ND2-1 Discrete Input module has been discontinued as of 07/2021: Please consider the Productivity, BRX or CLICK Series of PLC systems as a replacement.

Errata Sheet

## This Errata Sheet contains corrections or changes made after the publication of this manual.

### 08.2018

## Changes to Chapter 2. Installation and Safety Guidelines

Page 2-12. Base Wiring
On newer 110-240 VAC bases, the terminal strip has been re-designed and does not have different terminals for 110VAC and 240VAC connections. If you have one of these re-designed bases the terminal strip will look like the graphic shown here.


Changes to Chapter 3. DL330/DL330P/DL340 CPU Specifications
Page 3-9. DL330/DL330P CPU Setup; Installing the UVPROM Option in the DL330/DL330P CPU.
The drawing showing the jumpers is missing the silkscreen jumper lines and arrows for jumpers 1, 2, and 3 . Refer to this drawing.



## This Errata Sheet contains corrections or changes made after the publication of this manual.

### 05.2018

## Changes to Chapter 3. DL330/DL330P/DL340 CPU Specifications

Page 3-14. Battery Backup:
Please revise the first WARNING on this page (right above the battery replacement drawings) as follows:

WARNING: If the battery is not installed or connected to the PC board, the 330 CPU will NOT notify you of the error. Be sure the battery is in place and the connector is firmly seated before you install the CPU into the base.

## Changes to Chapter 4. Bases, Expansion Bases, and I/O Configuration

Page 4-16. Setting the Base Switches; 5 Slot Bases:
Replace the drawing of the 5 slot base with this one. Newer 5 slot bases have jumper switch SW1 instead of the toggle switch to set whether the base is a local CPU base or an expansion base.

NOTE: Older bases have a toggle switch to set the base as the CPU local base, the first expansion base, or the second (last) espansion base. Newer bases have the jumper SW1 in place of the switch.


Jumper SW1


## Changes to Chapter 5. I/O Module Selection \& Wiring Guidelines

## Page 5-12. Fuse Protection

Replace the WARNING on this page with the following one:

WARNING: Modules which have soldered-in fuses or non-replaceable fuses are non-repairable and should be replaced with new modules.

Errata Sheet

# This Errata Sheet contains corrections or changes made after the publication of this manual. 

### 05.2018, cont'd

## Changes to Chapter 11. Instruction Set; Timer, Counter, and Shift Register Instructions

Page 11-22. Timer (TMR) DL330/DL340 Only
Page 11-23. Counter (CNT) DL330/DL340 Only
Add this NOTE to both of these pages:
NOTE: The counters and timers both time in Decimal and not in BCD. Presets for both are also interpreted as decimal data and not as BCD.

## Changes to Chapter 13. Maintenance and Troubleshooting

Page 13-13. Add the following to the end of this chapter (right after END Instruction Placement):

## Reset the PLC to Factory Defaults

NOTE: Resetting to factory defaults will not clear any password stored in the PLC.

Resetting a DirectLogic PLC to Factory Defaults is a two-step process. Be sure to have a verified backup of your program using "Save Project to Disk" from the File menu before performing this procedure. Please be aware that the program as well as any settings will be erased and not all settings are stored in the project. In particular you will need to write down any settings for Secondary Communications Ports and manually set the ports up after resetting the PLC to factory defaults.

Step 1 - While connected to the PLC with DirectSoft, go to the PLC menu and select; "Clear PLC Memory". Check the "ALL" box at the bottom of the list and press "OK".
Step 2 - While connected with DirectSoft, go the PLC menu and then to the "Setup" submenu and select "Initialize Scratch Pad". Press "Ok".

NOTE: All configurable communications ports will be reset to factory default state. If you are connected via Port 2 or another configurable port, you may be disconnected when this operation is complete.

## NOTE: Retentive ranges will be reset to the factory settings.

NOTE: Manually addressed IO will be reset to factory default settings.
The PLC has now been reset to factory defaults and you can proceed to program the PLC.



Warning! Intializing the scratchpad memory resets the operating parameters of the PLC back to their factory default settings.

Program memory is retained but user defined parameters such as secondary communications port settings, manual 1/0 configuration, retentive range modification, etc. will be reset to their factory defaults.

If you have changed any of these parameters you must reconfigure these settings after initializing the scratchpad.

Initialize Scratch Pad Memory?
OK Cancel
Help

Errata Sheet

This Errata Sheet contains corrections or changes made after the publication of this manual.
06.13.2012

Changes to Chapter 3. DL330/DL330P/DL340 CPU Specifications
Page 3-3. CPU Specifications
DL340 CPUs DO NOT support overrides. In row 3, Supports Overrides, change the "Yes" in the DL340 column to "No".

# VAUTOMATIONDIRECTE! 

## DL305 User Manual

Manual Number: D3-USER-M

D3-350 CPU has been retired as of 12/2021. PLease consider BRX, CLICK, or Productivity systems as a replacement.
D3-340 CPU retired as of 11/2021.
D3-330 CPU retired as of 8/26/2021. Please consider D3-340 or D3-350 CPU as a replacement.

Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

Thank you for purchasing automation equipment from Automationdirect.com ${ }^{T M}$, doing business as, AutomationDirect. We want your new automation equipment to operate safely. Anyone who installs or uses this equipment should read this publication (and any other relevant publications) before installing or operating the equipment.

To minimize the risk of potential safety problems, you should follow all applicable local and national codes that regulate the installation and operation of your equipment. These codes vary from area to area and usually change with time. It is your responsibility to determine which codes should be followed, and to verify that the equipment, installation, and operation are in compliance with the latest revision of these codes.

At a minimum, you should follow all applicable sections of the National Fire Code, National Electrical Code, and the codes of the National Electrical Manufacturer's Association (NEMA). There may be local regulatory or government offices that can also help determine which codes and standards are necessary for safe installation and operation.

Equipment damage or serious injury to personnel can result from the failure to follow all applicable codes and standards. We do not guarantee the products described in this publication are suitable for your particular application, nor do we assume any responsibility for your product design, installation, or operation.

Our products are not fault-tolerant and are not designed, manufactured or intended for use or resale as on-line control equipment in hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines, or weapons systems, in which the failure of the product could lead directly to death, personal injury, or severe physical or environmental damage ("High RiskActivities"). AutomationDirect specifically disclaims any expressed or implied warranty offitness for High Risk Activities.

For additional warranty and safety information, see the Terms and Conditions section of our Desk Reference. If you have any questions concerning the installation or operation of this equipment, or if you need additional information, please call us at 770-844-4200.

This publication is based on information that was available at the time it was printed. We at AutomationDirect constantly strive to improve our products and services, so we reserve the right to make changes to the products and/or publications at any time without notice and without any obligation. This publication may also discuss features that may not be available in certain revisions of the product.

## Trademarks

This publication may contain references to products produced and/or offered by other companies. The product and company names may be trademarked and are the sole property of their respective owners. AutomationDirect disclaims any proprietary interest in the marks and names of others.

## Copyright 2010, Automationdirect.com ${ }^{\text {™ }}$ Incorporated All Rights Reserved

No part of this manual shall be copied, reproduced, or transmitted in any way without the prior, written consent of Automationdirect.com Incorporated. AutomationDirect retains the exclusive rights to all information included in this document.

Nous vous remercions d'avoir acheté l'équipement d'automatisation de Automationdirect.comMC, en faisant des affaires comme, AutomationDirect. Nous tenons à ce que votre nouvel équipement d'automatisation fonctionne en toute sécurité. Toute personne qui installe ou utilise cet équipement doit lire la présente publication (et toutes les autres publications pertinentes) avant de l'installer ou de l'utiliser.

Afin de réduire au minimum le risque d'éventuels problèmes de sécurité, vous devez respecter tous les codes locaux et nationaux applicables régissant l'installation et le fonctionnement de votre équipement. Ces codes diffèrent d'une région à l'autre et, habituellement, évoluent au fil du temps. Il vous incombe de déterminer les codes à respecter et de vous assurer que l'équipement, l'installation et le fonctionnement sont conformes aux exigences de la version la plus récente de ces codes.

Vous devez, à tout le moins, respecter toutes les sections applicables du Code national de prévention des incendies, du Code national de l'électricité et des codes de la National Electrical Manufacturer's Association (NEMA). Des organismes de réglementation ou des services gouvernementaux locaux peuvent également vous aider à déterminer les codes ainsi que les normes à respecter pour assurer une installation et un fonctionnement sûrs.

L'omission de respecter la totalité des codes et des normes applicables peut entraîner des dommages à l'équipement ou causer de graves blessures au personnel. Nous ne garantissons pas que les produits décrits dans cette publication conviennent à votre application particulière et nous n'assumons aucune responsabilité à l'égard de la conception, de l'installation ou du fonctionnement de votre produit.

Nos produits ne sont pas insensibles aux défaillances et ne sont ni conçus ni fabriqués pour l'utilisation ou la revente en tant qu'équipement de commande en ligne dans des environnements dangereux nécessitant une sécurité absolue, par exemple, l'exploitation d'installations nucléaires, les systèmes de navigation aérienne ou de communication, le contrôle de la circulation aérienne, les équipements de survie ou les systèmes d'armes, pour lesquels la défaillance du produit peut provoquer la mort, des blessures corporelles ou de graves dommages matériels ou environnementaux ("activités à risque élevé"). La société AutomationDirect nie toute garantie expresse ou implicite d'aptitude à l'emploi en ce qui a trait aux activités à risque élevé.

Pour des renseignements additionnels touchant la garantie et la sécurité, veuillez consulter la section Modalités et conditions de notre documentation. Si vous avez des questions au sujet de l'installation ou du fonctionnement de cet équipement, ou encore si vous avez besoin de renseignements supplémentaires, n'hésitez pas à nous téléphoner au 770-844-4200.

Cette publication s'appuie sur l'information qui était disponible au moment de l'impression. À la société AutomationDirect, nous nous efforçons constamment d'améliorer nos produits et services. C'est pourquoi nous nous réservons le droit d'apporter des modifications aux produits ou aux publications en tout temps, sans préavis ni quelque obligation que ce soit. La présente publication peut aussi porter sur des caractéristiques susceptibles de ne pas être offertes dans certaines versions révisées du produit.

## Marques de commerce

La présente publication peut contenir des références à des produits fabriqués ou offerts par d'autres entreprises. Les désignations des produits et des entreprises peuvent être des marques de commerce et appartiennent exclusivement à leurs propriétaires respectifs. AutomationDirect nie tout intérêt dans les autres marques et désignations.

## Copyright 2010, Automationdirect.com Incorporated Tous droits réservés

Nulle partie de ce manuel ne doit être copiée, reproduite ou transmise de quelque façon que ce soit sans le consentement préalable écrit de la société Automationdirect.com Incorporated. AutomationDirect conserve les droits exclusifs à l'égard de tous les renseignements contenus dans le présent document.

## Manual History

Refer to this history in all correspondence and/or discussion about this manual.
Title: DL305 User Manual
Manual Number: D3-USER-M

| Issue | Date | Description of Changes |
| :--- | :---: | :--- |
| Original | $1 / 94$ | Original Issue |
| Rev A | $8 / 95$ | Made corrections throughout |
| Rev A-1 | $10 / 96$ | Manual resized, update of selected pages <br> issued in Rev A |
| Rev B | $5 / 98$ | Made minor revisions before reprinting |
| Rev C | $8 / 02$ | Replaced F3-16TA-1 with F3-16TA-2 |
| Rev D | $1 / 2010$ | Made corrections throughout manual |

F3-04DA-1 has been retired as of 11/29/2023.
D3-16TD2 has been retired as of 06/2022.
D3-16TD1-1 has been retired as of 05/2022.
D3-05B-1 and D3-08TA-2 have been retired as of 09/2021.
Please consider Productivity, BRX or CLICK as a replacement PLC platform.
Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

## Table of Contents

## Chapter 1: Getting Started

Introduction ..... 1-2
The Purpose of this Manual ..... 1-2
Who Should Read this Manual ..... 1-2
Where to Begin ..... 1-2
Supplemental Manuals ..... 1-2
How this Manual is Organized ..... 1-3
Technical Assistance ..... 1-3
DL305 System Components ..... 1-4
DirectSOFT Programming for Windows ${ }^{\text {TM }}$ ..... 1-4
Handheld Programmer ..... 1-4
DL305 System Diagrams ..... 1-4
DirectLOGIC ${ }^{\text {TM }}$ Part Numbering System ..... 1-8
Steps to a Successful System ..... 1-10
Step 1: Review the Installation Guidelines ..... 1-10
Step 2: Understand the CPU Setup Procedures ..... 1-10
Step 3: Understand the I/O System Configurations ..... 1-10
Step 4: Review the I/O Selection Criteria ..... 1-10
Step 5: Determine the I/O Module Specifications and Wiring Characteristics ..... 1-10
Step 6: Understand the System Operation ..... 1-11
Step 7: Review the Programming Concepts ..... 1-11
Step 8: Choose the Instructions ..... 1-11
Step 9: Understand the Maintenance and Troubleshooting Procedures ..... 1-11
Chapter 2: Installation and Safety Guidelines
Safety Guidelines ..... 2-2
Plan for Safety ..... 2-2
Safety Techniques ..... 2-2
Orderly System Shutdown ..... 2-3
System Power Disconnect ..... 2-3
Panel Design Specifications ..... 2-4
Environmental Specifications ..... 2-6
Power ..... 2-6
Agency Approvals ..... 2-7
Enclosures ..... 2-7
Component Dimensions ..... 2-7
Component Dimensions Part 2 ..... 2-9
Base Mounting Dimensions ..... 2-10
Installing Components in the Base ..... 2-10
Base Wiring ..... 2-11
Expansion Base Wiring ..... 2-11
I/O Wiring ..... 2-12
I/O Wiring Guidelines ..... 2-12
Wiring the Different Module Types ..... 2-13
Chapter 3: DL330/DL330P/DL340 CPU Specifications
Overview ..... 3-2
DL330 CPU Features ..... 3-2
DL330P CPU Features ..... 3-2
DL340 CPU Features ..... 3-2
CPU Hardware Features ..... 3-2
CPU Specifications ..... 3-3
Selecting CPU Memory Options ..... 3-4
Internal Retentive Memory ..... 3-4
External Program Storage ..... 3-4
Volatile and Non-volatile Memory ..... 3-4
Program Storage Memory Types (Internal) ..... 3-5
Storing Programs on UVPROMs ..... 3-6
Setting up the PROM Writer Unit ..... 3-7
Copying a Program From the CPU RAM to a UVPROM ..... 3-7
Copying a Program From the UVPROM to the CPU RAM ..... 3-8
Comparing a Program From the UVPROM to the CPU RAM ..... 3-8
Erasing a UVPROM ..... 3-8
DL330/DL330P CPU Setup ..... 3-9
Installing the UVPROM Option in the DL330 / DL330P CPU ..... 3-9
Selecting Retentive Memory for the DL330 / DL330P ..... 3-9
DL330/DL330P Networking ..... 3-9
DL340 CPU Setup ..... 3-10
Installing the optional UVPROM or EEPROM in the DL340 CPU ..... 3-10
Selecting Retentive Memory for the DL340 ..... 3-11
DL340 Port Setup ..... 3-12
DL340 Baud Rate Selection ..... 3-12
DL340 Network Address Selection ..... 3-12
DL340 RS232C Port (1 and 2) Pin Outs ..... 3-13
DL340 Station Type Selection and Address Ranges ..... 3-13
DL340 Selecting the Response Delay Time ..... 3-13
DL340 Selecting Data Format (ASCII/HEX) ..... 3-13
DL340 Selecting Parity for Port 2 ..... 3-13
Battery Backup ..... 3-14
Memory Battery Backup ..... 3-14
DL330, DL330P, DL340 CPU Battery Replacement ..... 3-14
Installing the CPU ..... 3-15
CPU Setup and System Functions ..... 3-16
A Few Things to Know ..... 3-16
What are Auxiliary Functions? ..... 3-17
Connecting the Programming Devices ..... 3-18
Changing the CPU Mode of Operation ..... 3-20
Clearing the CPU Memory ..... 3-21
CPU Checklist ..... 3-21
Chapter 4: Bases, Expansion Bases, and I/O Configuration
Understanding I/O Numbering and Module Placement Rules ..... 4-2
DL305 I/O Configuration History ..... 4-2
Octal Numbering System ..... 4-2
Fixed I/O Numbering ..... 4-2
I/O Numbering Guidelines ..... 4-3
Number of I/O Points Required for Each Module ..... 4-3
I/O Module Placement Rules ..... 4-4
DL330/DL330P Rules for 16 Point Modules ..... 4-5
DL340 Rules for 16 Point Modules ..... 4-6
Base Specifications and Wiring ..... 4-7
Three Sizes of Bases ..... 4-7
Bases and Maximum I/O Supported ..... 4-8
Base Mounting Dimensions ..... 4-8
Connecting the Power Supply ..... 4-9
Expansion Base Power Supply Wiring Example ..... 4-9
Base Specifications ..... 4-10
Auxiliary 24VDC Output at Base Terminal ..... 4-10
Power Supply Schematics ..... 4-11
Using the Run Relay on the Base Power Supply ..... 4-12
Installing CPUs and I/O Modules ..... 4-13
Using Bases for Local or Expansion I/O Systems ..... 4-14
Base Uses Table ..... 4-14
Local/Expansion Connectivity ..... 4-14
Connecting Expansion Bases ..... 4-15
Setting the Base Switches ..... 4-16
5 Slot Bases ..... 4-16
10 Slot Base ..... 4-16
Example I/O Configurations ..... 4-17
16 Point I/O Allocation Example ..... 4-17
Examples Show Maximum I/O Points Available ..... 4-17
I/O Configurations with a 5 Slot Local CPU Base ..... 4-18
Switch settings ..... 4-18
5 Slot Base with 8 Point I/O ..... 4-18
5 Slot Base with 16 Point I/O ..... 4-18
5 Slot Base and 5 Slot Expansion Base with 8 Point I/O ..... 4-19
5 Slot Base and 5 Slot Expansion Base with 16 Point I/O ..... 4-19
5 Slot Base and Two 5 Slot Expansion Bases with 8 Point I/O ..... 4-20
5 Slot Base and Two 5 Slot Expansion Bases with 16 and 8 Point I/O ..... 4-20
I/O Configurations with an 8 Slot Local CPU Base ..... 4-21
8 Slot Base with 8 Point I/O ..... 4-21
8 Slot Base with 16 Point I/O ..... 4-21
8 Slot Base and 5 Slot Expansion Base with 8 Point I/O ..... 4-21
8 Slot Base and 5 Slot Expansion Base with 16 Point I/O ..... 4-21
I/O Configurations with a 10 Slot Local CPU Base ..... 4-22
Switch settings ..... 4-22
Last Slot Address Range 100 to 107 ..... 4-22
Last Slot Address Range 700 to 707 ..... 4-22
10 Slot Expansion Base with 16 Point I/O ..... 4-23
Configuration 1 ..... 4-23
Configuration 2 ..... 4-23
10 Slot Base and 5 Slot Expansion Base with 16 Point I/O ..... 4-24
Expansion Addresses Depend on Local CPU Base Configuration. ..... 4-25
10 Slot Base and 10 Slot Expansion Base with 8 Point I/O ..... 4-25
10 Slot Base and 10 Slot Expansion Base with 16 Point I/O ..... 4-25
Calculating the Power Budget ..... 4-26
Managing your Power Resource ..... 4-26
Auxiliary Base Power Source ..... 4-26
Base Power Specifications ..... 4-26
Module Power Requirements ..... 4-27
Power Budget Calculation Example ..... 4-30
Power Budget Calculation Worksheet ..... 4-31
Chapter 5: I/O Module Selection \& Wiring Guidelines
I/O Selection Considerations ..... 5-2
I/O Module Selection ..... 5-2
Sinking and Sourcing Circuits ..... 5-2
DL305 Input Module Configuration Chart ..... 5-3
DL305 Output Module Configuration Chart ..... 5-3
Configuration \#1 DL305 DC Current Sourcing Input Module ..... 5-4
Configuration \#2 DL305 DC Current Sinking/Sourcing Input Module ..... 5-4
Configuration \#3 DL305 DC Current Sinking Input Module ..... 5-5
Configuration \#4 DL305 AC/DC Input Module ..... 5-5
Configuration \#5 DL305 AC Input Module ..... 5-6
Configuration \#6 DL305 DC Current Sinking Output Module ..... 5-7
Configuration \#7 DL305 DC Current Sourcing Output Module ..... 5-7
Configuration \#8 DL305 AC/DC Current Sink/Source (Relay) Output Module ..... 5-8
Configuration \#9 DL305 AC Output Module ..... 5-8
Solid State Field Device Wiring to DC Input Modules ..... 5-9
NPN Field Device Example ..... 5-9
PNP Field Device Example ..... 5-9
Derating Characteristics ..... 5-10
I/O Wiring Guidelines ..... 5-11
General Considerations ..... 5-11
Wiring the Different Module Types ..... 5-11
Fuse Protection ..... 5-12
External Fuse Example ..... 5-12

## Chapter 6: Discrete Input Modules

Discrete Input Module Identification and Terminology ..... 6-2
Discrete Input Module Status Indicators ..... 6-2
Color Coding of I/O Modules ..... 6-2
Input Module Selection ..... 6-2
Inputs Per Module ..... 6-3
Commons Per Module ..... 6-3
Input Voltage Range ..... 6-3
Peak Voltage ..... 6-3
AC Frequency ..... 6-3
ON Voltage Level ..... 6-3
OFF Voltage Level ..... 6-3
Input Current ..... 6-3
Input Impedance ..... 6-3
Minimum ON Current ..... 6-3
Maximum OFF Current ..... 6-3
Base Power Required ..... 6-3
OFF to ON Response ..... 6-3
ON to OFF Response ..... 6-3
Terminal Type ..... 6-3
Status Indicators ..... 6-3
Weight ..... 6-3
D3-08ND2, 24 VDC Input Module ..... 6-4
D3-16ND2-1, 24 VDC Input Module ..... 6-5
D3-16ND2-2, 24 VDC Input Module Module ..... 6-6
D3-16ND2F, 24 VDC Fast Response Input Module ..... 6-7
F3-16ND3F, TTL/24 VDC Fast Response Input Module ..... 6-8
D3-08NA-1, 110 VAC Input Module ..... 6-10
D3-08NA-2, 220 VAC Input Module ..... 6-11
D3-16NA, 110 VAC Input Module ..... 6-12
D3-08NE3, 24 VAC/DC Input Module ..... 6-13
D3-16NE3, 24 VAC/DC Input Module ..... 6-14
D3-08SIM, Input Simulator ..... 6-15
Chapter 7: Discrete Output Modules
Discrete Output Module Identification and Terminology ..... 7-2
Discrete Output Module Status Indicators ..... 7-2
Color Coding of I/O Modules ..... 7-2
Output Modules Selection ..... 7-2
Outputs Per Module ..... 7-3
Commons Per Module ..... 7-3
Operating Voltage ..... 7-3
Output Type ..... 7-3
Peak Voltage ..... 7-3
AC Frequency ..... 7-3
ON Voltage Drop ..... 7-3
Maximum Current (Resistive) ..... 7-3
Maximum Leakage Current ..... 7-3
Maximum Inrush Current ..... 7-3
Minimum Load ..... 7-3
Base Power Required ..... 7-3
OFF to ON Response Time ..... 7-3
ON to OFF Response Time ..... 7-3
Terminal Type ..... 7-3
Status Indicators ..... 7-3
Fuses ..... 7-3
Relay Life ..... 7-3
Weight ..... 7-3
Relay Arc Suppression - DC and AC Applications ..... 7-4
FL305 High Current Relay Output Module Arc Suppression ..... 7-4
Resistor and Capacitor Selection ..... 7-4
Resistor Tolerance ..... 7-4
Peak Voltage and Current ..... 7-4
Adding Contact Protection ..... 7-4
Resistor and Capacitor Nomogram ..... 7-5
D3-08TD1, 24 VDC Output Module ..... 7-6
D3-08TD2, 24 VDC Output Module ..... 7-7
D3-16TD1-1, 24 VDC Output Module ..... 7-8
D3-16TD1-2, 24 VDC Output Module ..... 7-9
D3-16TD2, 24 VDC Output Module ..... 7-10
D3-04TAS, 110-220 VAC Output Module ..... 7-11
F3-08TAS, 250 VAC Isolated Output Module ..... 7-12
F3-08TAS-1, 125 VAC Isolated Output Module ..... 7-13
D3-08TA-1, 110-220 VAC Output Module ..... 7-14
D3-08TA-2, 110-220 VAC Output Module ..... 7-15
F3-16TA-2, 20-125 VAC Output Module ..... 7-16
D3-16TA-2, 110-220 VAC Output Module ..... 7-17
D3-08TR, Relay Output Module ..... 7-18
F3-08TRS-1, Relay Output Module ..... 7-19
F3-08TRS-2, Relay Output Module ..... 7-20
D3-16TR, Relay Output Module ..... 7-21
Chapter 8: System Operation
Introduction ..... 8-2
CPU Operating System ..... 8-3
DL305 CPU Operational Flow Chart ..... 8-3
Initial Mode Setting and Memory Initialization ..... 8-4
Flow Chart for Initial Mode Setting (\#1) ..... 8-4
Memory Initialization ..... 8-5
Program Mode Operation ..... 8-6
Run Mode Operation ..... 8-7
Update I/O Points ..... 8-8
Service Peripherals ..... 8-9
Service for Monitoring and Forcing Operations ..... 8-10
Solve Application Program ..... 8-12
I/O Response Time ..... 8-14
Is Timing Important for Your Application? ..... 8-14
Normal Minimum I/O Response ..... 8-14
Normal Maximum I/O Response ..... 8-15
Improving Response Time ..... 8-15
CPU Scan Time Considerations ..... 8-16
DL330 / DL330P Scan Calculation ..... 8-16
DL340 Scan Calculation ..... 8-17
Application Program Execution ..... 8-18
Memory Map ..... 8-19
Octal Numbering System ..... 8-19
Two Basic Memory Types: Discrete and Word ..... 8-19
R Memory Locations for Discrete Memory Areas ..... 8-20
I/O Points ..... 8-20
Control Relays ..... 8-21
Timers and Timer Status Bits (T Data type) ..... 8-22
Counters and Counter Status Bits (CT Data type) ..... 8-22
Counter Current Values (R Data Type) ..... 8-22
Data Registers (R Data Type) ..... 8-23
Stages (S Data type) ..... 8-23
Shift Registers ..... 8-24
Special Relays ..... 8-24
Special Registers (R Data Type) ..... 8-24
DL305 Aliases ..... 8-25
DL330 Memory Map ..... 8-26
DL330P Memory Map ..... 8-27
DL340 Memory Map ..... 8-28
I/O Point Bit Map ..... 8-29
Control Relay Bit Map ..... 8-30
Special Relays ..... 8-32
Timer / Counter Registers and Contacts ..... 8-33
External Timer/Counter Setpoint Unit ..... 8-33
Data Registers ..... 8-34
Stage Control / Status Bit Map ..... 8-36
Shift Register Bit Map ..... 8-37
Special Registers ..... 8-38

## Chapter 9: Programming Basics

Introduction ..... 9-2
Using Boolean Instructions ..... 9-3
END Statement ..... 9-3
Simple Rungs ..... 9-3
Normally Closed Contact ..... 9-3
Contacts in Series ..... 9-4
Midline Outputs ..... 9-4
Parallel Elements ..... 9-4
Joining Series Branches in Parallel ..... 9-5
Joining Parallel Branches in Series ..... 9-5
Comparative Boolean ..... 9-5
Combination Networks ..... 9-6
Boolean Stack ..... 9-6
Using Timers ..... 9-8
Using Counters ..... 9-9
Using the Accumulator ..... 9-10
Copying Data to and from the Accumulator ..... 9-10
Changing the Accumulator Data ..... 9-11
Accumulator Operations ..... 9-12
Chapter 10: RLLPLUS Programming Basics
Introduction ..... 10-2
An Example Machine ..... 10-3
Machine Operation ..... 10-3
Machine Flowchart ..... 10-3
An RLL Solution ..... 10-4
An RLL ${ }^{\text {PLUS }}$ Solution ..... 10-5
Stage Instruction Execution ..... 10-6
Stage Instruction Numbering ..... 10-6
A Few Simple Rules for Execution ..... 10-7
Activating Stages ..... 10-8
Using Initial Stages ..... 10-8
Using Jump Instructions ..... 10-9
Using Set Instructions with Stages ..... 10-10
Power Flow Transitions ..... 10-11
Using Outputs in Stages ..... 10-12
Setting Outputs with the SET Instruction ..... 10-12
Using the OUT Instruction ..... 10-13
Latching Outputs with Stages ..... 10-14
Using Timers and Counters in Stages ..... 10-15
Time Based Transitions ..... 10-15
Using Counters ..... 10-16
Using Data Instructions in Stages ..... 10-17
Using Comparative Contacts in Stages ..... 10-18
Parallel Branching Concepts ..... 10-19
Branching Methods ..... 10-19
Joining Parallel Branches ..... 10-20
Using Stage Bits as Contacts ..... 10-21
Stage Contact Example ..... 10-22
Unusual Operations in Stages ..... 10-23
Using the Same Output Multiple Times ..... 10-23
Using a Set Out Reset (SET OUT RST) Instruction ..... 10-24
Output Placement ..... 10-24
Two Ways to View RLLPLUS Programs ..... 10-25
DirectSOFT Stage View ..... 10-25
DirectSOFT Ladder View ..... 10-25
Designing a Program Using RLLPLUS Instructions ..... 10-27
Use DirectSOFT to Save Time ..... 10-27
Step 1: Design a Top-level Flowchart ..... 10-28
Step 2: Add Flowchart Transitions ..... 10-29
Step 3: Add Actions ..... 10-30
Step 4: Add Conditions for Actions ..... 10-31
Step 5: Add Alarm or Monitoring Operations ..... 10-32
Step 6: Determine Stage Numbering ..... 10-33
Step 7: Assign I/O Addresses ..... 10-35
Step 8: Enter the Program ..... 10-36
Chapter 11: Instruction Set
Introduction ..... 11-2
Store (STR) ..... 11-4
Store Not (STR NOT) ..... 11-4
Store Timer (STR TMR) DL330/340 Only ..... 11-5
Store Not Timer (STR NOT TMR) DL330/340 Only ..... 11-5
Store Counter (STR CNT) DL330/340 Only ..... 11-6
Store Not Counter (STR NOT CNT) DL330/340 Only ..... 11-6
Or (OR) ..... 11-7
Or Not (OR NOT) ..... 11-7
Or Timer (OR TMR) DL330/340 Only ..... 11-8
Or Not Timer (OR NOT TMR) DL330/340 Only ..... 11-8
Or Counter (OR CNT) DL330/340 Only ..... 11-9
Or Not Counter (OR NOT CNT) DL330/340 Only ..... 11-9
And (AND) ..... 11-10
And Not (AND NOT) ..... 11-10
And Timer (AND TMR) DL330/340 Only ..... 11-11
And Not Timer (AND NOT TMR) DL330/340 Only ..... 11-11
And Counter (AND CNT) DL330/340 Only ..... 11-12
And Not Counter (AND NOT CNT) DL330/340 Only ..... 11-12
And Store (AND STR) ..... 11-13
Or Store (OR STR) ..... 11-13
Out (OUT) ..... 11-15
Set (SET) DL330/340 Only ..... 11-16
Reset (RST) DL330/340 Only ..... 11-16
Set Out (SET OUT) ..... 11-17
Set Out Reset (SET OUT RST) ..... 11-18
Store If Equal (STR) DL330/DL340 Only ..... 11-19
Store Not If Equal (STR NOT) DL330/DL340 Only ..... 11-19
Or If Equal (OR) DL330/DL340 Only ..... 11-20
Or Not If Equal (OR NOT) DL330/DL340 Only ..... 11-20
And If Equal (AND) DL330/DL340 Only ..... 11-21
And Not If Equal (AND NOT) DL330/DL340 Only ..... 11-21
Timer (TMR) DL330/DL340 Only ..... 11-22
Counter (CNT) DL330/DL340 Only ..... 11-23
Shift Register (SR) DL330/340 Only ..... 11-24
Data Store DSTR (F50) ..... 11-25
Data Out DOUT (F60) ..... 11-25
Data Store 1 DSTR (F51) ..... 11-26
Data Out 1 DOUT (F61) ..... 11-26
Data Store 2 DSTR (F52) ..... 11-27
Data Out 2 DOUT (F62) ..... 11-27
Data Store 3 DSTR (F53) ..... 11-28
Data Out 3 DOUT (F63) ..... 11-28
Data Store 5 DSTR (F55) ..... 11-29
Data Out 5 DOUT (F65) ..... 11-29
Data And DAND (F75) ..... 11-30
Data Or DOR (F76) ..... 11-31
Compare CMP (F70) ..... 11-32
Add ADD (F71) ..... 11-34
Add Example ..... 11-35
Subtract SUB (F72) ..... 11-36
Subtract Example ..... 11-37
Multiply MUL (F73) ..... 11-38
Multiply Example ..... 11-39
Divide DIV (F74) ..... 11-40
Divide Example ..... 11-41
Shift Left SHFL (F81) ..... 11-42
Shift Right SHFR (F80) ..... 11-43
Number Conversion Instructions ..... 11-44
Encode ENCOD (F83) ..... 11-44
Decode DECOD (F82) ..... 11-46
Binary BIN (F85) ..... 11-47
Binary Coded Decimal BCD (F86) ..... 11-48
Invert INV (F84) ..... 11-49
Program Control Instructions ..... 11-50
Master Control Set (MCS) and Master Control Reset (MCR) DL330/DL340 only ..... 11-50
Understanding Master Control Relays ..... 11-50
MCS/MCR Example ..... 11-51
Network Instructions ..... 11-52
Read from Network RX (F952) DL340 Only ..... 11-52
Write to Network WX (F953) DL340 Only ..... 11-54
Message Instructions ..... 11-56
Fault FAULT (F20) ..... 11-56

## Chapter 12: RLLPLuS Instruction Set

Introduction ..... 12-2
Initial Stage (ISG) DL330P Only ..... 12-3
Stage (SG) DL330P Only ..... 12-3
Jump (JMP) DL330P Only ..... 12-5
Not Jump (NOT JMP) DL330P Only ..... 12-5
Store Stage (STR SG) DL330P Only ..... 12-7
Store Not Stage (STR NOT SG) DL330P Only ..... 12-7
Or Stage (OR SG) DL330P Only ..... 12-8
Or Not Stage (OR NOT SG) DL330P Only ..... 12-8
And Stage (AND Stage) DL330P Only ..... 12-9
And Not Stage (AND NOT SG) DL330P Only ..... 12-9
Set (SET) DL330P Only ..... 12-10
Reset (RST) DL330P Only ..... 12-10
Set Stage (SET SG) DL330P Only ..... 12-11
Reset Stage (RST SG) DL330P Only ..... 12-11
Comparative Boolean Instructions ..... 12-12
Store If Greater Than or Equal To Timer (STR TMR) DL330P Only ..... 12-12
Store Not IF Greater Than Timer (STR NOT TMR) DL330P Only ..... 12-12
Store If Greater Than or Equal To Counter (STR CNT) DL330P Only ..... 12-13
Store Not If Greater Than Counter (STR NOT CNT) DL330P Only ..... 12-13
Or If Greater Than or Equal To Timer (OR TMR) DL330P Only ..... 12-14
Or Not If Greater Than Timer (OR NOT TMR) DL330P Only ..... 12-14
Or If Greater Than or Equal To Counter (OR CNT) DL330P Only ..... 12-15
Or Not If Greater Than Counter (OR NOT CNT) DL330P Only ..... 12-15
And If Greater Than or Equal To Timer (AND TMR) DL330P Only ..... 12-16
And Not If Greater Than Timer (AND NOT TMR) DL330P Only ..... 12-16
And If Greater Than or Equal To Counter (AND CNT) DL330P Only ..... 12-17
And Not If Greater Than Counter (AND NOT CNT) DL330P Only ..... 12-17
Timer, Counter, and Shift Register Instructions ..... 12-18
Timer (TMR) DL330P Only ..... 12-18
Counter (CNT) DL330P Only ..... 12-19
Reset Counter (RST) DL330P Only ..... 12-20
Shift Register (SR) DL330P Only ..... 12-21
Chapter 13: Maintenance and Troubleshooting
Maintenance ..... 13-2
CPU Indicators ..... 13-3
Power Indicator ..... 13-4
RUN Indicator ..... 13-6
CPU Indicator ..... 13-6
BATT Indicator ..... 13-6
Expansion Base Power ..... 13-7
Testing Output Points ..... 13-8
I/O Module Troubleshooting ..... 13-10
Noise Troubleshooting ..... 13-11
Machine Startup and Program Troubleshooting ..... 13-12
Appendix A: Quick Start Example
Step 1: Unpack the DL305 Equipment ..... A-2
Step 2: Configure the 5-slot Base as the Local CPU Base ..... A-3
Step 3: Install the CPU and I/O Modules ..... A-3
Step 4: Wire the I/O Modules to the Field Devices ..... A-4
Step 5: Remove the Terminal Strip Access Cover ..... A-4
Step 6: Connect the Power Wiring ..... A-5
Step 7: Connect the Handheld Programmer ..... A-6
Step 8: Connect the Power Source ..... A-7
Step 9: Enter the Example Program ..... A-7
Appendix B: DL305 Error Codes
Appendix C: Instruction Execution Times
Introduction ..... C-2
Data Registers ..... C-2
I/O Data Registers ..... C-2
How to Read the Tables ..... C-3
DL330 Instruction Execution Times ..... C-4
Basic Input Instructions ..... C-4
Output Type Instructions ..... C-4
Timer, Counters, and Shift Registers ..... C-5
Data Operations ..... C-5
DL330P Instruction Execution Times ..... C-6
Basic Input Instructions ..... C-6
Output Type Instructions ..... C-6
Timer, Counters, and Shift Registers ..... C-6
Stage Instructions ..... C-7
Data Operation Instructions ..... C-7
DL340 Instruction Execution Times ..... C-8
Basic Input Instructions ..... C-8
Comparative Contacts ..... C-8
Output Type Instructions ..... C-8
Timer, Counters, and Shift Registers ..... C-9
Data Operation Instructions ..... C-9
Appendix D: DL305 Product Weight Tables
Product Weight Table ..... D-2

## Index

## Getting Started

In This Chapter. . . .

- Introduction
- DL305 System Components
— DirectLOGIC ${ }^{\text {T" }}$ Part Numbering System
- A Few Steps to a Successful System


## Introduction

The Purpose of
this Manual

Thank you for purchasing our DL305 family of automation products. This manual shows you how to install the equipment, and it also helps you understand the system operation characteristics.


Since we constantly try to improve our product line, we occasionally issue addenda that document new features and changes to the products. If an addendum is included with this manual, please read it to see which areas of the manual or product have changed.

## Who Should Read this Manual

## Where to Begin

## Supplemental Manuals

If you understand PLC systems our manuals will provide all the information you need to get and keep your system up and running. We will use examples and explanations to clarify our meaning and perhaps help you brush up on specific features used in the DL305 system. This manual is not intended to be a generic PLC training manual, but rather a user reference manual for the DL305 system.

If you are in a hurry and already understand the DL305 system please read Chapter 2, Installation and Safety Guidelines, and proceed on to the chapter pertaining to your needs. Be sure to keep this manual handy for reference when you run into questions. If you are a new DL305 customer, we suggest you read this manual completely so you can understand the wide variety of products, configurations, and procedures used with the DL305 family of products. We believe you will be pleasantly surprised with how much you can accomplish with AutomationDirect products.
If you're really in a hurry, check out Appendix A . This appendix has a quick start that will show you how to quickly connect and program a very simple system.

Depending on the products you have purchased, there may be other manuals that are necessary for your application. If you have purchased analog I/O, specialty modules, or DirectSOFT, or you will be using remote I/O or networking, you will want to supplement this manual with the manuals written for these products.

How this Manual is Ch 1: Getting Started - provides an overview of all the components that can be Organized

Technical
Assistance used to make up one or many DL305 systems. This chapter shows the basic concepts of how the pieces fit together. It also explains the DL305 part numbering system, which will help you quickly identify the various types of modules.
Ch 2: Installation and Safety Guidelines - shows you how to prepare for system installation, and gives you guidelines for providing a safe environment for your personnel and process. Be sure to read this chapter so potential safety problems can be avoided. In this chapter you will find topics you must consider when installing a system, the environmental specifications, component dimensions, safety guidelines, installation guidelines, etc.
Ch 3: DL330/DL330P/DL340 CPU Specifications - provides details of each of the DL305 CPUs. This chapter contains the operating specifications for the CPUs, detailed information on the different types of program storage media available, and some basic procedures needed to get the CPU ready for programming.
Ch 4: System Configuration, Bases and Expansion Bases - provides selection and installation criteria for Local I/O and Local Expansion I/O. This chapter also discusses the system power budget, which is an important part of the planning and installation process.
Ch 5: I/O Module Selection Criteria - contains specific considerations which affect I/O selection such as sinking, sourcing, and temperature derating characteristics.
Ch 6: Discrete Input Modules - explains each term you will find on our specification sheets, provides specifications, wiring diagrams and derating curves (where applicable) for the DL305 Discrete Input Modules.
Ch 7: Discrete Output Modules - explains each term you will find on our specification sheets, provides specifications, wiring diagrams and derating curves (where applicable) for the DL305 Discrete Output Modules.
Ch 8: System Operation - explains how the DL305 CPUs control the system operation. This includes information on I/O updates, application program execution and memory structure.
Ch 9 : RLL Programming Concepts - explains the basic concepts used in RLL programming.
Ch 10: RLL PLUS Programming Concepts - explains the basic concepts used in the RLLPLUS programming. This programming method greatly reduces program design time and simplifies machine startup and troubleshooting.
Ch 11: Instruction Set - explains how each individual instruction operates.
Ch 12: RLLPLUS Instruction Set - explains the instructions used with the DL330P CPU. It also shows some instructions that operate differently with this CPU.
Ch 13: Maintenance and Troubleshooting - is a guide designed to aid you in diagnosing, repairing and avoiding system problems.
Appendices A - D - there are several appendices referred to throughout the manual. These include things such as a quick start, error code listing, instruction execution times, etc.
We realize even though we strive to be the best, we may have arranged our information in such a way you cannot find what you are looking for. If you need assistance, please, call us at 1-800-633-0405. Our technical support group is glad to work with you in answering your questions. They are available weekdays from 8:00 a.m. to 6:00 p.m. eastern standard time. If you find a problem with any of our products, services or manuals, please fill out and return the Suggestions card included with this manual.

# DL305 System Components 

CPUs

Bases
I/O Configuration

## I/O Modules

## Programming Methods

## DirectSOFT

Programming for Windows ${ }^{\text {™ }}$

Handheld
Programmer

DL305 System Diagrams

The DL305 product family is one of the most versatile and widely accepted PLCs used for small control applications. These CPUs are small yet powerful. Their modular design and expansion capability blend well with todays fast moving industry. The following is a summary of the major DL305 system components.

There are three CPUs in this product line, the DL330, the DL330P and the new DL340. Details of these CPU are covered in Chapter 3, DL330/DL330P/DL340 CPU Specifications.

Three base sizes are available in the system: 5 slot, 8 slot and 10 slot.
The DL330 and DL330P CPUs support up to 128 local I/O and 176 local expansion I/O. The DL340 supports 136 local I/O and 184 local expansion I/O. Each of these I/O configurations is explained in Chapter 4, Bases and Expansion Bases and I/O Configuration.

The DL305 has one of the most diverse I/O module selections in the industry. A complete range of discrete modules which support 24 VDC, 125 VDC, 110/220 VAC and up to 10A relay outputs are offered. The analog modules provide 12 bit resolution and several selections of input and output signal ranges (including bipolar). The specialty modules include 10 KHz high speed input, thermocouple, general purpose communication, and more.

There are two programming methods available, RLL (Relay Ladder Logic) and RLLPLUS. RLLPLUS combines the added feature of flow chart programming (stages) to the standard RLL language. RLLPLUS is only available for the DL330P CPU. All of the DL305 CPUs support RLL programming. DirectSOFT supports both RLL and RLLPLUS programming. Two handheld programmers are available, the D3-HPP which supports RLLPLUS and the D3-HP which only supports RLL programming. The key pads for each handheld programmer differ, so it is recommended the handheld programmer that directly supports your CPU be used for programming.

The DL305 can be programmed with one of the most advanced programming packages in the industry - DirectSOFT. DirectSOFT runs under Windows and supports many of the windows based features you are already familiar with such as cut and paste between applications, point and click editing, viewing and editing multiple application programs at the same time, browsers, etc. DirectSOFT universally supports the DirectLOGIC CPU families. This means you can use the same DirectSOFT package to program DL205, DL305, DL405 or any new CPUs we add to our product line. There is a separate manual that discusses DirectSOFT programming software.

All DL305 CPUs have a built-in programming port for use with the handheld programmers (D3-HPP and D3-HP). Handheld programmers can be used to create, modify and store programs to cassette tape, as well as debug your application program. There is also a separate manual that discusses the DL305 Handheld Programmers.

The next page shows a generic example highlighting the major components and configurations of the DL305 system. The following two pages highlight the specific components which can be used to build your system.

## Machine Control

Packaging
Conveyors
Elevators
Handheld Programmer



## Computer Controlled I/O

Industry Standard Computer I/O Protocol
OPTOMUX ${ }^{\text {TM }}$ (Serial RS422/485)
PAMUX ${ }^{\text {M }}$ (Parallel)


Computer Interface with OPTOMUX ${ }^{\text {TM }}$ Driver


## DirectLOGIC <br> DL305 Family



AC/DC INPUT
8pt 24 VAC/DC 16pt 24 VAC/DC




## NETWORKING

RS232C Data Communication Unit RS422 Data Communication Unit MODBUS® Slave Module MODBUS® Slave Module w/Radio Modem

Universal connector:
RS232C / RS422/485 Convertor

## DirectLOGIC Part Numbering System

As you examine this manual, you'll notice there are many different products available. Sometimes it is difficult to remember the specifications for any given product. However, If you take a few minutes to understand the numbering system, it may save you some time and confusion. The charts below show how the part numbering systems work for each product category. Part numbers for accessory items such as cables, batteries, memory cartridges etc. are typically an abbreviation of the description for the item.


|  |  | D4- | 16 | N | D | 2 | F |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Discrete I/O |  | D3- | 16 | N | D | 2 | -1 |  |
| DL205 Product family | D2/F2 |  |  |  |  |  | , |  |
| DL305 Product family | D3/F3 |  |  |  |  |  | ' |  |
| DL405 Product family | D4/F4 |  |  |  |  |  | , |  |
| Number of points | 04/08/12/16/32 |  |  |  |  |  | ' |  |
| Input | N |  |  |  |  |  | ' |  |
| Output | T |  |  |  |  |  | , |  |
| Combination | C |  |  |  |  |  | ' |  |
| AC | A |  |  |  |  |  | , |  |
| DC | D |  |  |  |  |  | ! |  |
| Either | E |  |  |  |  |  | ' |  |
| Relay | R |  |  |  |  |  | , |  |
| Current Sinking | 1 |  |  |  |  |  | : |  |
| Current Sourcing | 2 |  |  |  |  |  | ', |  |
| Current Sinking/Sourcing | 3 |  |  |  |  |  | , |  |
| High Current | H |  |  |  |  |  |  |  |
| Isolation | S |  |  |  |  |  | ' |  |
| Fast I/O | F |  |  |  |  |  |  |  |
| Denotes a differentiation between Similar modules | -1, -2, -3, -4 |  |  |  |  |  |  |  |

## DirectLOGIC Part Numbering System (cont.)



Altermate example of Analog I/O using abbreviations
note: -n indicates thermocouple type such as: J, K, T, R, S or E

| Communication and Networking, |  |
| :--- | :--- |
| Special I/O and Devices |  |
| Programming | D2/F2 |
| DL205 Product family | D3/F3 |
| DL305 Product family | D4/F4 |
| DL405 Product family | see example |
| Name Abbreviation |  |




## A Few Steps to a Successful System

Step 1: Review the Installation Guidelines

Step 2: Understand the CPU Setup Procedures

Step 3:
Understand the
I/O System
Configurations

Step 4:
Review the I/O Selection Criteria

Step 5:
Determine the I/O
Module
Specifications and Wiring Characteristics


You should always make safety your first priority in any system application. Chapter 2 provides several guidelines that will help provide a safer, more reliable system. This chapter also includes wiring guidelines for the various system components.

The CPU is the heart of your automation system. Make sure you take the time to understand the various features and setup requirements.

It is important to understand how the I/O system can be configured. You have two different types of systems.

- Local System
- Local Expansion System

It is also important to understand how the system Power Budget is calculated. This can affect your I/O placement and/or configuration options.

There are many considerations involved when you select your I/O modules. Take time to understand how the various types of electrical connections can affect your choice of I/O modules.

There are many different I/O modules available with the DL305 system. Chapters 6 and 7 provide the specifications and wiring diagrams for the discrete I/O modules.

NOTE: Specialty modules have their own manuals and are not included in this manual


Step 6:
Understand the System Operation

Step 7:
Review the
Programming
Concepts

Step 8:
Choose the Instructions

Step 9:
Understand the Maintenance and
Troubleshooting Procedures

Before you begin to enter a program, it is very helpful to understand how the DL305 system processes information. This involves not only program execution steps, but also involves the various modes of operation and memory layout characteristics.

All control systems differ in some areas. The DL305 CPUs offer two different types of programming. RLL programming available for all the DL305 CPUs, uses conventional ladder diagram type solutions for many application problems.
RLLPLUS is available for the DL330P CPU. This method of programming greatly reduces the program design time and makes program troubleshooting and machine startup considerably easier.

Once you have installed the system and understand the theory of operation, you can choose from a diverse instruction set to implement your application.

Many things can happen on the factory floor. Switches fail, batteries need to be replaced, etc. In most cases, the majority of the troubleshooting and maintenance time is spent trying to locate the problem. Chapter 13 provides some information that will help you quickly identify problems, so you can look like a hero if you take time to understand them.



## Installation and Safety Guidelines

In This Chapter. . . .

- Safety Guidelines
- Panel Design Specifications
- Component Dimensions
- Base Mounting Dimensions
- Installing Components in the Base
- I/O Wiring


## Safety Guidelines


#### Abstract

WARNING: Providing a safe operating environment for personnel and equipment is your responsibility and should be your primary goal during system planning and installation. Automation systems can fail and may result in situations that can cause serious injury to personnel or damage to equipment. Do not rely on the automation system alone to provide a safe operating environment. You should use external electromechanical devices, such as relays or limit switches, that are independent of the PLC system to provide protection for any part of the system that may cause personal injury or damage.


Every automation application is different, so there may be special requirements for your particular application. Make sure you follow all National, State, and local government requirements for the proper installation and use of your equipment.

Plan for Safety The best way to provide a safe operating environment is to make personnel and equipment safety part of the planning process. You should examine every aspect of the system to determine which areas are critical to operator or machine safety.
If you are not familiar with PLC system installation practices, or your company does not have established installation guidelines, you should obtain additional information from the following sources.

- NEMA - The National Electrical Manufacturers Association, located in Washington, D.C., publishes many different documents that discuss standards for industrial control systems. You can order these publications directly from NEMA. Some of these include:
ICS 1, General Standards for Industrial Control and Systems
ICS 3, Industrial Systems
ICS 6, Enclosures for Industrial Control Systems
- NEC - The National Electrical Code provides regulations concerning the installation and use of various types of electrical equipment. Copies of the NEC Handbook can often be obtained from your local electrical equipment distributor or your local library.
- Local and State Agencies - many local governments and state governments have additional requirements above and beyond those described in the NEC Handbook. Check with your local Electrical Inspector or Fire Marshall office for information.

Safety Techniques The publications mentioned provide many ideas and requirements for system safety. At a minimum, you should follow these regulations. Also, you should use the following techniques, which may help reduce the risk of safety concerns.

- Orderly system shutdown sequence in the PLC control program.
- System power disconnects (guard limits, emergency stop switches, etc.)

Three Levels of Protection

The publications mentioned provide many ideas and requirements for system safety. At a minimum, you should follow these regulations. Using the techniques listed below will further help reduce the risk of safety problems.

- Emergency stop switch for disconnecting system power.
- Mechanical disconnect for output module power.
- Orderly system shutdown sequence in the PLC control program.

Emergency Stops It is recommended that emergency stop circuits be incorporated into the system for every machine controlled by a PLC. For maximum safety in a PLC system, these circuits must not be wired into the controller, but should be hardwired external to the PLC. The emergency stop switches should be easily accessed by the operator and are generally wired into a master control relay (MCR) or a safety control relay (SCR) that will remove power from the PLC I/O system in an emergency.
MCRs and SCRs provide a convenient means for removing power from the I/O system during an emergency situation. by de-energizing an MCR (or SCR) coil, power to the input (optional) and output devices is removed. This event occurs when any emergency stop switch opens. However, the PLC continues to receive power and operate even though all its inputs and outputs are disabled.
The MCR circuit could be extended by placing a PLC fault relay (closed during normal PLC operation) in series with any other emergency stop conditions. This would cause the MCR circuit to drop the PLC I/O power in case of a PLC failure (memory error, I/O communications error. etc.).


Emergency Power A properly rated emergency power disconnect should be used to power the PLC Disconnect

## Orderly System

 Shutdown controlled system as ameans of removing the power from the entire control system. It may be necessary to install a capacitor across the disconnect to protect against a condition known as "outrush". This condition occurs when the output triacs are turned off by powering off the disconnect, thus causing the energy stored in the inductive loads to seek the shortest distance to ground, which is often through the triacs.After an emergency shutdown or any other type of power interruption, there may be requirements that must be met before the PLC control program can be restarted. For example, there may be specific register values that must be established (or maintained from the state prior to the shutdown) before operations can resume. In this case, you may want to use retentive memory locations, or include constants in the control program to ensure a known starting point.

The first level of protection should be included in the PLC control program, which can be used to identify machine problems. You should analyze your application and identify any shutdown sequences that must be performed. These types of problems are usually things such as jammed parts, etc. that do not pose a risk of personal injury or equipment damage.
WARNING: The control program
should not be the only form of
protection for any problems that may
result in a risk of personal injury or
equipment damage.


Class 1, Division 2 This equipment is suitable for use in Class 1, Division 2, groups A, B, C and D or Approval non-hazardous locations only.

WARNING: Explosion Hazard! Substitution of components may impair suitability for Class 1, Division 2.
Do not disconnect equipment unless power has been switched off or area is known to be non-hazardous.

## Panel Design Specifications

It is important to design your panel properly to help ensure the DL305 products operate within their environmental and electrical limits. Proper installation of your DL305 system requires an in-depth understanding of electrical control systems. The system installation should comply with the appropriate electrical codes and standards for your area. It is important that your system also conforms to the operating standards for the application to insure proper performance. The DL305 equipment should only be installed by personnel familiar with electrical/industrial applications. The DL305 installation should provide proper ventilation, spacing, and grounding to ensure the equipment will operate as specified. The diagram on the next page references the items in the following list.

1. The bases should be mounted horizontally to provide proper ventilation.
2. There should be a minimum of $7.2^{\prime \prime}$ ( 183 mm ) and a maximum of 13.75 " ( 350 mm ) between bases.
3. A minimum clearance of 2 " ( 50 mm ) between the base and the top, bottom and right side of the cabinet should be provided.
4. A minimum clearance of 3 " ( 75 mm ) between the base and the left side of the cabinet should be provided.
5. There must be a minimum of 2 " clearance between the panel door and the nearest DL305 component.
6. The ground terminal on the DL305 base must be connected to a single point ground. Copper stranded wire should be used for this connection to achieve a low impedance. Copper eye lugs should be crimped and soldered to the ends of the stranded wire to assure good surface contact. You should also remove anodized finishes and use copper lugs and star washers at termination points. A rule of thumb is to achieve a 0.1 ohm of DC resistance between the DL305 base and the single point of ground.
7. There must be a single point of ground (i.e. copper bus bar) for all devices in the panel requiring an earth ground return. The single point of ground must be connected to the panel ground termination.
The panel ground termination must be connected to earth ground. For this connection you should use \#12 AWG stranded copper wire as a minimum. Minimum wire sizes, color coding, and general safety practices should comply with appropriate electrical codes and standards for your area.
A good common ground reference (Earth ground) is essential for proper operation of the DL305. The DL305 system and components are designed to operate with a common ground reference. There are several methods of providing an adequate common ground reference. These methods include:
a) Installing a ground rod as close to the panel as possible.
b) Connection to the incoming power system ground.
8. Installations where the ambient temperature may approach the lower or upper limits of the specifications should be evaluated properly. To do this place a temperature probe in the panel, close the door and operate the system until the ambient temperature has stabilized. If the ambient temperature is not within the operating specification for the DL305 system, measures such as installing a cooling/heating source must be taken to get the ambient temperature within the DL305 operating specifications.
9. Device mounting bolts and ground braid termination bolts should be \#10 copper bolts or equivalent. Tapped holes instead of nut-bolt arrangements should be used whenever possible. To assure good contact on termination areas impediments such as paint, coating or corrosion should be removed in the area of contact.
10. The DL305 systems are designed to be powered by 110 VAC , 220 VAC, or 24 VDC normally available throughout an industrial environment. Isolation transformers and noise suppression devices are not normally necessary, but may be helpful in eliminating/reducing suspected power problems.


In addition to the panel layout guidelines, there are other specifications that can affect the definition and installation of a PLC system. You should always consider the following areas whenever you install any PLC system.

- Environmental Specifications
- Power Supply Specifications
- Agency Approvals
- Enclosure Selection and Component Dimensions


## Environmental Specifications

Power The power source must be capable of suppling voltage and current complying with the base power supply specifications.

| Specifications | D3-05B | D3-05BDC | D3-08B | D3-10B |
| :---: | :---: | :---: | :---: | :---: |
| Input Voltage Range | $\begin{aligned} & 100-240 \text { VAC } \\ & +10 \% /-15 \% \\ & 47-63 \mathrm{~Hz} \end{aligned}$ | $\begin{aligned} & 20.5-30 \text { VDC }<10 \% \\ & \text { ripple } \end{aligned}$ | $\begin{array}{\|l} \hline 100-240 \text { VAC } \\ +10 \% /-15 \% \\ 47-63 \mathrm{~Hz} \end{array}$ | $\begin{aligned} & 100-240 \text { VAC } \\ & +10 \% /-15 \% \\ & 47-63 \mathrm{~Hz} \end{aligned}$ |
| Base Power Consumption | 70 VA max (46W) | 48 Watts | 70 VA max (57W) | 70 VA max (57W) |
| Inrush Current max. | 30A | 30A | 30A | 30A |
| Dielectric Strength | 1500VAC for 1 minute between terminals of AC P/S, Run output, Common, 24VDC | 1500VAC for 1 minute between 24VDC input terminals and Run output | 1500VAC for 1 minute between terminals of AC P/S, Run output, Common, 24VDC | 2000VAC for 1 minute between terminals of AC P/S, Run output, Common, 24VDC |
| Insulation Resistance | $>10 \mathrm{M} \Omega$ at 500VDC | $>10 \mathrm{M} \Omega$ at 500VDC | $>10 \mathrm{M} \Omega$ at 500VDC | $>10 \mathrm{M} \Omega$ at 500VDC |
| Power Supply Output (Voltage Ranges and Ripple) | $\begin{array}{\|l} \hline \text { (5VDC) } 4.75-5.25 \mathrm{~V} \\ \text { less than } 0.1 \mathrm{~V} \mathrm{p-p} \\ \text { ( } 9 \mathrm{VDC} \text { ) } 8.5-13.5 \mathrm{~V} \\ \text { less than } 0.2 \mathrm{~V} \mathrm{p}-\mathrm{p} \\ \text { (24VDC) } 20-28 \mathrm{~V} \\ \text { less than } 1.2 \mathrm{~V}-\mathrm{p} \end{array}$ | (5VDC) 4.75-5.25V less than 0.1 V p-p (9VDC) $8.5-13.5 \mathrm{~V}$ less than 0.2 V p-p (24VDC) 20-28V less than 1.2 V p-p | $\begin{aligned} & \text { (5VDC) } 4.75-5.25 \mathrm{~V} \\ & \text { less than } 0.1 \mathrm{~V} \mathrm{p-p} \\ & \text { ( } 9 \mathrm{VDC} \text { ) } 8.0-12.0 \mathrm{~V} \\ & \text { less than } 0.2 \mathrm{~V} \mathrm{p-p} \\ & \text { (24VDC) } 20-28 \mathrm{~V} \\ & \text { less than } 1.2 \mathrm{~V}-\mathrm{p} \end{aligned}$ | (5VDC) 4.75-5.25V less than 0.1 V p-p (9VDC) 8.0-12.0V less than 0.2 V p-p (24VDC) 20-28V less than 1.2 V p-p |

Agency Approvals Some applications require agency approvals. Typical agency approvals which your application may require are:

- UL (Underwriters' Laboratories, Inc.)
- CSA (Canadian Standards Association)
- FM (Factory Mutual Research Corporation)
- CUL (Canadian Underwriters' Laboratories, Inc.)


## Enclosures

Your selection of a proper enclosure is important to ensure safe and proper operation of your DL305 system. Applications of DL305 systems vary and may require additional features. The minimum considerations for enclosures include:

- Conformance to electrical standards
- Protection from the elements in an industrial environment
- Common ground reference
- Maintenance of specified ambient temperature
- Access to equipment
- Security or restricted access
- Sufficient space for proper installation and maintenance of equipment


## Component Dimensions

Before installing your PLC system you will need to know the dimensions for the components in your system. The diagrams on the following pages provide the component dimensions and should be used to define your enclosure specifications. Remember to leave room for potential expansion. Appendix D provides the weights for each component.


Component
Dimensions Part 2


## Base Mounting Dimensions

Below are the mounting dimensions which should be used when mounting DL305 bases. Make sure you have followed the installation guidelines for proper spacing.


## Installing Components in the Base

When inserting components into the base, align the PC board(s) of the module with the grooves on the top and bottom of the base. Push the module straight into the base until it is firmly seated in the backplane connector.


Base Wiring The following diagram shows the terminal connections located on the power supply of the DL305 bases.

WARNING: Damage will occur to the base power supply if 220 VAC is connected to the 115 VAC terminal connections. Once the power wiring is connected, install the protective cover. When the cover is removed there is a risk of electrical shock if you accidentally touch the connection terminals.


24 VDC
Terminal Strip


Expansion Base Wiring

This is an example of how to connect power when using local CPU and Expansion bases.


## I/O Wiring

This information provides a general idea on how to wire the different types of modules in the DL305 system. For specific information on wiring a particular module refer to the specification sheet in the appropriate I/O chapter or manual.

I/O Wiring Guidelines

You should consider these guidelines when wiring your system.

1. There is a limit to the size of wire the modules can accept. The table below lists the maximum AWG for each module type. Smaller AWG is acceptable to use for each of the modules.

| Module type | Maximum AWG |
| :--- | :--- |
| 8 point | 12 |
| 16 point | 16 |

2. Always use a continuous length of wire, do not combine wires to attain a needed length.
3. Use the shortest possible cable length.
4. Use wire trays for routing where possible
5. Avoid running wires near high energy wiring.
6. Avoid running input wiring in close proximity to output wiring where possible.
7. To minimize voltage drops when wires must run a long distance , consider using multiple wires for the return line.
8. Avoid running DC wiring in close proximity to AC wiring where possible.
9. Avoid creating sharp bends in the wires.

Wiring the Different There are three main types of module faces for the DL305 I/O. These module faces Module Types are: lift covers over terminal blocks, flip covers over terminal blocks and D-shell compatible sockets. If the module you are using has a cover you can remove the cover either by lifting from the bottom or by flipping the door open. Some of the modules have removable terminal blocks. These modules can be recognized by the squeeze tabs on the top and bottom of the terminal block. To remove the terminal block, press the squeeze tabs and pull the terminal block away from the module.


WARNING: For some modules, field device power may still be present on the terminal block even though the PLC system is turned off. To minimize the risk of electrical shock, check all field device power before you remove the connector.


## DL330/DL330P/DL340 CPU Specifications

In This Chapter. . . .

- Overview
- CPU Hardware Features
- CPU Specifications
- Selecting CPU Memory Options
- DL330/DL330P CPU Setup
- DL340 CPU Setup
— DL340 Port Setup
- Battery Backup
- Installing the CPU
- CPU Setup and System Functions

D3-340-EE chips discontinued with no replacements, 06/2023
D3-05B-1 5 slot base has been retired as of 09/2021. Please consider BRX, Productivity or CLICK systems as a replacement PLC platform.
D3-330 CPU has been retired as of 8/26/2021. Please consider D3-340 or D3-350 CPU as a replacement.
Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

DL330 CPU Features

DL330P CPU Features

DL340 CPU Features

The CPU is the heart of the control system. Almost all system operations are controlled by the CPU, so it is important that it is set-up and installed correctly. This chapter provides the information needed to understand:

- the differences between the different models of CPUs
- the different memory options
- the steps required to setup and install the CPU.

The DL330 modular CPU is capable of controlling $176 \mathrm{I} / \mathrm{O}$ points and has 3.7 K words of program storage. This CPU supports the RLL programing language and can save programs internally to RAM or UVPROM. There is a built-in programming port that directly supports the handheld programmer.
The DL330P modular CPU is capable of controlling $176 \mathrm{I} / \mathrm{O}$ points and has 3.7 K words program storage. This CPU supports the RLLPLUS programing language and can save programs internally to RAM or UVPROM. RLLPLUS provides a structured programming environment for Relay Ladder Logic through the addition of stage logic. There is a built-in port that directly supports the handheld programmer.

The DL340 modular CPU is capable of controlling $184 \mathrm{I} / \mathrm{O}$ points and has 3.7 K words program storage. This CPU supports the RLL programming language and can save programs internally to RAM, UVPROM or EEPROM. There is a handheld programming port and two built-in RS232C ports for PC programming, operator interfaces, or networking. If you are using the DL340 in a DirectNET network, you can use either port as a slave port and the bottom port as a master. The bottom port has the additional capability of being configured as a slave on a Modbus® network.

## CPU Hardware Features

| CPU Status Indicators |  |  |
| :--- | :--- | :--- |
| RUN | ON <br> OFF | CPU is in RUN mode <br> CPU is in Program mode |
| BATT | ON <br> OFF | Memory backup voltage low <br> Memory backup voltage good |
| CPU | ON | CPU failure (Error detected when the <br> watchdog timer is not processed within <br> 100ms. The run output from the power <br> supply will be turned off.) <br> CPU good |
| PWR | ON | CPU power good <br> CPU power failure |
| OF | ON | CPU communication port <br> receiving data <br> CPU communication port <br> not receiving data |
| TX | OFF | CPU communication port <br> transmitting data <br> CPU communication port <br> not transmitting data |



CPU Specifications
See Errata sheet on the first page. DL330 Retired.

See Errata Sheet on the first page. DL340 CPUs DO NOT support overrides.

| Feature | DL330 | DL330P | DL340 |
| :---: | :---: | :---: | :---: |
| Program memory (words) | 3.7K | 3.7K | 3.7K |
| Scan time/K ladder (boolean) | 8 ms | 20 ms | . 87 ms |
| Supports Overrides | No | No | Yes No |
| RLL (Relay Ladder Logic) Programming | Yes | Yes | Yes |
| RLL ${ }^{\text {PLUS }}$ Programming | No | Yes | No |
| Handheld programmer with cassette tape interface | Yes | Yes | Yes |
| DirectSOFT programming for Windows ${ }^{\text {mM }}$ | Yes | Yes | Yes |
| Built-in communication ports (RS232C / DirectNET) | No | No | Yes |
| CMOS RAM | Yes | Yes | Yes |
| UVPROM | Optional | Optional | Optional |
| EEPROM | No | No | Optional |
| Compatible with: |  |  |  |
| ASCII Basic modules | Yes | Yes | Yes |
| Networking modules | Yes | Yes | Yes |
| RS232C Data Communications Unit | Yes | Yes | Yes |
| RS422 Data Communications Unit | Yes | Yes | Yes |
| Base Power Supply Available |  |  |  |
| 110/220 VAC | Yes | Yes | Yes |
| 24 VDC (5 slot base only) | Yes | Yes | Yes |
| Total I/O points using; |  |  |  |
| Local I/O | 128 | 128 | 136 |
| Local expansion I/O | 176 | 176 | 184 |
| Remote I/O | NA | NA | NA |
| Number of instructions available | 61 | 65 | 61 |
| Control relays | 140 | 77 | 196 |
| Shift register bits | 128 | uses CRs | 128 |
| Special relays (system defined) | 12 | 11 | 20 |
| Stages (RLL ${ }^{\text {PLUS }}$ only) | None | 128 | None |
| Timer/Counters | 64 | 64 | 64 |
| Data registers | 128 | 128 | 192 |
| Analog input channels max. | 112 | 112 | 128 |
| Analog output channels max. | 28 | 28 | 32 |
| Internal diagnostics | Yes | Yes | Yes |
| Password security | Yes | Yes | Yes |
| Battery backup | Yes | Yes | Yes |

## Selecting CPU Memory Options

Internal Retentive Memory

External Program Storage

Volatile and
Non-volatile Memory

In addition to different choices for program storage, you can also select some memory areas to be retentive. Retentive memory retains its state after a power cycle or a program to run transition occurs, as long as the memory backup battery is functional. Non-retentive memory resets to a logical " 0 " after a power cycle or a program to run transition occurs. You have to use dipswitch to select the retentive memory options (the switches are discussed in the next section.)
The following table shows the how the types of memory are defined. Some types of memory are automatically defined as retentive and other memory types can be defined as retentive as necessary for your application. The types of memory available depend on the type of CPU selected for your application.

| Retentive Memory | Pre-defined | User defined |
| :--- | :--- | :--- |
| Application program | Yes |  |
| Stages (DL330P only) |  | Yes |
| Internal relays |  | Yes |
| Current count values | Yes (full range) |  |
| Shift register bits | Yes (full range) |  |
| Data registers | Yes (full range) |  |
| Password |  | Yes |

All DL305 CPUs allow for program storage to be captured on external media such as cassette tape, floppy disk and hard disk. Refer to the DL305 Handheld Programmer manual for details on storing the CPU program to cassette tape. The DirectSOFT manual provides details on storing the CPU program to floppy or hard disk.

There are two types of memory storage available, volatile and non-volatile. Volatile memory will retain your data as long as proper voltage is maintained to the storage media. Non-volatile memory does not require power to retain data. The DL305 CPUs maintain the proper voltage either through the base power supply or the use of the memory backup battery.

Program Storage
Memory Types (Internal)

The type of program storage memory available to you depends on the CPU you are using. All DL305 CPUs support application program storage to either CMOS RAM or the optional UVPROM. The DL340 has the added option of supporting program storage in EEPROM. The application program can be up to 3.7 K words.

- CMOS RAM memory (Random Access Memory) is standard on all the DL305 CPUs. It is a volatile memory which can be modified or changed easily with a handheld programmer or PC programming software.
- UVPROM (Ultraviolet Programmable Read Only Memory) is optional for all the DL305 CPUs. This type of memory is non-volatile and can only be erased with an ultraviolet light source. The PROM Writer Unit (D3-PWU) is used to copy your application program from the CPU's RAM to a UVPROM. If the UVPROM has a program to be changed, it must be removed from the CPU and erased before another program can be copied on the UVPROM.
- EEPROM (Electrically Erasable Programmable Read Only Memory) is an option only on the DL340 CPU. This type of memory is non-volatile, but can be electrically erased. The EEPROM can be electrically reprogrammed without being removed from the CPU, and without the use of a special external programming device.

WARNING: Be sure to use proper grounding techniques when touching UVPROMS and EEPROMS. A static discharge from you may cause damage to the PROM. If you do not have a ground strap, then ground yourself by touching the controller chassis before you make contact with the PROM. Also ensure that the surface where you place the PROM is properly grounded.

## Storing Programs on UVPROMs

The PROM Writer Unit is only compatible with DL305 CPUs and UVPROMs. It can perform the following three functions:

- Copy a program from the CPU's RAM to a UVPROM
- Copy a program from the UVPROM to the CPU's RAM
- Compare the program in the UVPROM with the CPU's RAM


The LED for the selected function will turn off when completed (except for the error reset function). If any error is encountered, one of the LEDs in the following table will be on and the execution of the selected function will be stopped.
$\left.\begin{array}{|l|l|l|l|l|}\hline \text { Function } & \begin{array}{l}\text { Key } \\ \text { Operation }\end{array} & \begin{array}{l}\text { LED } \\ \text { Display }\end{array} & \text { Remarks } & \text { Errors Flagged } \\ \hline \begin{array}{l}\text { Copies the content of } \\ \text { the CPU RAM into the } \\ \text { UVPROM }\end{array} & \text { WRITE } & \text { OWRITE } & \begin{array}{l}\text { Automatic comparison } \\ \text { is made after } \\ \text { checking and writing. }\end{array} & \begin{array}{l}\text { Constant on indicates } \\ \text { a write failure. }\end{array} \\ \hline \begin{array}{l}\text { Copies the content of } \\ \text { the UVPROM into the } \\ \text { CPU RAM }\end{array} & \begin{array}{l}\text { WRITE } \\ \text { VERIFY }\end{array} & \text { OWRITE } \\ \text { OVERIFY }\end{array} \begin{array}{l}\text { Depress two keys at } \\ \text { the same time. } \\ \text { Comparison is made } \\ \text { after transferring. }\end{array}\right]$.

## Setting up the PROM Writer Unit

The PROM Writer Unit connects directly to either a DL330, DL330P or DL340 CPU. Use the following steps to connect the PROM Writer Unit:

1. Set the power supply source switch (on the back of the unit) to the appropriate power source setting, (INT for using base power and EXT for an external power source). The PROM Writer Unit can either use the local CPU base power or use an external power source.

NOTE: If you are using the local CPU base power you will need to include the Prom Writer Unit power consumption in your power budget. The power budget is covered in Chapter 4.
2. If using an external power source attach the supplied cable to the power source socket on the back of the unit. The white wire should be connected to +5 VDC and the black wire should be connected to DC ground.
3. Turn off the power source to the base before attaching the PROM Writer Unit.
4. Attach the PROM Writer Unit to the CPU. The connector on the back of the unit will mate with the programming port (PRG) of the CPU. Tighten the fixture screw to secure the two units together.
5. Apply power to the local CPU base and if necessary to the PROM Writer Unit. Once the PWR LED is on it will take approximately 10 seconds for the unit to initialize. During this time keystrokes will not be recognized.
Copying a Program The following steps explain how to copy a program from the CPU RAM to a From the CPU RAM UVPROM: to a UVPROM

1. Turn power on.
2. Raise the UVPROM socket lever.
3. Insert the UVPROM (notch up) in the socket and lower the lever.
4. Press the "WRITE" button. The following sequence of events will take place:

- The WRITE LED will turn on then off.
- The BLANK LED will come on. (This notes the checking sequence to ensure that the UVPROM is blank has started.)
- The BLANK LED will turn off and the WRITE LED will turn on.
- The WRITE LED will turn off and the VERIFY LED will turn on. (This indicates that the write is complete. While the VERIFY LED is on, a comparison between the UVPROM and the CPU RAM is being made.)
- The VERIFY LED will turn off. (This indicates the end of the copying function.)
- If an error has been detected, the ERR LED will come on. If this happens press the "ERR" key to clear the error and the "WRITE" key to repeat the procedure. If this does not correct the problem, repeat the procedure using a different UVPROM.

5. Turn power off, raise the UVPROM socket lever and remove UVPROM.

Copying a Program The following steps explain how to copy a program from the UVPROM to the CPU From the UVPROM RAM: to the CPU RAM 1. Turn power on.
2. Raise the UVPROM socket lever.
3. Insert the UVPROM (notch up) in the socket and lower the lever.
4. Simultaneously press "WRITE" and "VERIFY" buttons. The following sequence of events will take place:

- The BLANK, WRITE and VERIFY LEDs will all come on momentarily.
- The WRITE LED turns off.
- The VERIFY LED will stay on till the operation is completed.
- If an error has been detected, the ERR LED will come on. If this happens, press the "ERR" key to clear the error and repeat step 4.

5. Turn power off, raise the UVPROM socket lever and remove UVPROM.

Comparing a Program From the UVPROM to the CPU RAM

The following steps show how to compare a UVPROM program to the CPU RAM:

1. Turn power on.
2. Raise the UVPROM socket lever.
3. Insert the UVPROM (notch up) in the socket and lower the lever.
4. Press the "VERIFY" button. The following sequence of events will take place:

- The VERIFY LED indicator will come on.
- If verification is successful, the VERIFY LED will go off.
- If there is an error in the comparison the VERIFY LED will remain on.

5. Turn power off, raise the UVPROM socket lever and remove UVPROM.

Erasing a UVPROM UVPROMS can be erased through exposure to an ultraviolet light source. Make sure that the window to the UVPROM is not covered so that it may receive full exposure to the light source. A typical exposure would be: $12,000 \mu \mathrm{w} / \mathrm{cm}^{2}$ lamp @ 2.5 cm for 15-20 minutes.

## DL330/DL330P CPU Setup

Installing the UVPROM Option in the DL330 / DL330P CPU

- Disconnect the power from the base and allow approximately 60 seconds for the capacitor to discharge before removing the CPU.
- Disconnect the battery wires from the CPU.
- Remove the RAM chip from IC socket.
- Align the UVPROM notch with the IC socket notch on the CPU card.
- Carefully insert the UVPROM in the IC socket.
- Set dip switch 2 and Jumpers 1-3 for UVPROM (ROM).
- Reconnect the battery wires to CPU.

|  | RAM | UVPROM (ROM) |
| :---: | :---: | :---: |
| JUMPER 1 | 0 0 0 | - |
| JUMPER 2 | $\begin{aligned} & 0 \\ & 0 \\ & 0 \end{aligned}$ | - |
| JUMPER 3 | 000 | 000 |
| DIPSWITCH 2 | $\square 4^{(0 n)}$ | $\square \downarrow$ (off) |

This drawing does not show the silkscreen jumper lines and arrows for jumpers 1, 2 and 3. See the Errata Sheet at the beginning of this file for an updated drawing.


Selecting Retentive The DL330 and DL330P have a dipswitch which can be used to turn on or off power Memory for the DL330 / DL330P

DL330/DL330P Networking
failure retention for specific relays and stages. (Some memory types are automatically retentive.) The following diagram lists the range of retentive memory for the memory types that are covered by the selection switch.

Networking for the DL330 and DL330P is accomplished by using a DCU, (Data Communications Unit, RS232C part number D3-232-DCU, RS422 part number D3-422-DCU).


Internal relays in the DL330 range from $160-373$, only 340-373 can be set retentive or non-retentive.

Internal relays in the DL330P range from 160-277, only 200-277 can be set retentive or non-retentive.
Stages in the DL330P range from
SG000 to SG177, only SG000 to SG137 can be set retentive or non-retentive.

## DL340 CPU Setup

Installing the optional UVPROM or EEPROM in the DL340 CPU

Complete the following steps to install the optional memory.

1. Disconnect the power from the base and allow approximately 60 seconds for the capacitor to discharge before removing the CPU.
2. Disconnect the battery wires from the CPU.
3. Align the UVPROM/EEPROM notch with the IC socket notch on the CPU.
4. Carefully insert the UVPROM/EEPROM into the IC socket.
5. Set dipswitch SW1, bit 1 and the short Jumpers N/C - 4 for the option you have installed.
6. Reconnect the battery wires to the CPU.

| Memory Type Switch | RAM | EEPROM | EEPROM (WRITE PROTECTED | UVPROM (ROM) |
| :---: | :---: | :---: | :---: | :---: |
| DIPSWITCH <br> SW1 - Bit 1 |  |  |  |  |
| SHORT PIN JUMPERS | $\begin{aligned} & \mathrm{N} / \mathrm{C} 2 \mathrm{l}_{4} \\ & \mathrm{O} 0 \mathrm{O} \end{aligned}$ | $\begin{aligned} & \mathrm{N} / \mathrm{C} 234 \\ & \mathrm{O} \begin{array}{llll} \mathrm{O} & \mathrm{O} & \mathrm{O} \end{array} \end{aligned}$ | $\begin{array}{cccc} \mathrm{N} / \mathrm{C} & 2 & 3 & 4 \\ \hline \mathrm{O} & \mathrm{O} & \mathrm{O} & \mathrm{O} \end{array}$ | $\begin{array}{cccc} \mathrm{N} / \mathrm{C} & 2 & 3 & 4 \\ \mathrm{O} & \mathrm{O} & \mathrm{O} & 0 \end{array}$ |



Selecting Retentive The DL340 uses the same dipswitch for selecting memory retention as was used for

Memory for the DL340 memory type selection. Dipswitch SW1, bit 2 is used to set memory retention for the ranges of internal relays shown in the following diagram.


## DL340 Port Setup

## DL340 Baud Rate Selection

The following chart shows how to configure the baud rate for Port 1 (RS232C) of the DL340 using dipswitch SW1, switches 3, 4 and 5. Port 2 baud rate is set by using a programming device to enter the baud rate in address R773 (in BCD or HEX).

## DL340 Network Address Selection



Port 2

| Baud | R773 |
| ---: | :---: |
| 300 | 1 |
| 600 | 2 |
| 1200 | 3 |
| 2400 | 4 |
| 4800 | 5 |
| 9600 | 6 |
| 19200 | 7 |
| 38400 | 8 |
| 9600 | 0,9 to $F F$ |

Sample Setup Ladder Logic



Port 1 (RS232C): Network address selection is accomplished with the Network Address Mode Switch and the two rotary switches 3 and 4. The address is set in BCD.
Network Address Mode Switch sets fixed or selectable network address.
Rotary Switch 3 sets the least significant decimal digit of the network address. Rotary Switch 4 sets the most significant decimal digit of the network address. In the example above, when the Network mode switch is set to FIXED the network address will default to 01, when the Network mode switch is set to USER the network address (set with the rotary switches) is 03 . Note, if the rotary switches are set to 00 , the network address will default to 01.

Port 2 (RS232C): Network address selection is set by using a programming device to enter the value for the most significant digit and least significant digit in addresses R771 and R 772 respectively. The address is set in BCD.
If you're using MODBUS RTU protocol on Port 2, the MODBUS address is set in decimal, not BCD. Load the lower two digits in R771 and the upper two digit(s) in R772.

DL340 RS232C
Port (1 and 2)
Pin Outs


| RS232C Communication Port Specifications |  |
| :---: | :---: |
| Connector ----------------RJ11 (handset connector) |  |
| Network Address | -01 to 90 |
|  |  |
| Parity -------Transfer Mode | -- - None / Odd |
|  | -Hex / ASCII |
|  | Half-duplex |
|  | Asynchronous |
| Data bits | - 8 |
| Start bits - | --1 |
| Stop bits | --1 |
| Turn Around Delay | -- - 0 to 1980 in 20 ms intervals (preset with R777) |

The station type for Port 1 is fixed as a Slave and cannot be changed. The station type for Port 2 can be selected by setting the appropriate switch positions (6 and 7) on the SW1 switch bank.


You can use the Handheld Programmer or DirectSOFT to select an on and off response delay time of up to 1980 ms . The time delay is calculated based on a preset number that is loaded into two memory locations. These presets indicate the number of 20 ms intervals that will be used as the delay. For example, an entry of 2 would result in a 40 ms response delay time.


DL340 Selecting Data Format (ASCII/HEX)

## DL340 Selecting Parity for Port 2

A special propose control relay is used to select between ASCII and HEX transmission modes on the DirectNET network. When this relay is off, HEX mode is used. When this relay is turned on, ASCII mode is used. Off is the default state.

- Port 1
C1077
- Port 2
C1076

DL340 CPUs with firmware V2.7 or later allow you to select the parity for Port 2. The default setting is none. A special propose control relay (C1072) is used to select
the Response Delay Time between odd parity (relay is on) and no parity (relay is off).

- Port 2

C1072

# Battery Backup 

Memory Battery
Backup Backup at he beginning of this file. This warning has been revised: WARNING: If the battery is not installed or connected to the PC board, the 330 CPU will NOT notify you of the error. Be sure the battery is in place and the connector is firmly seated before you install the CPU into the base.

DL330, DL330P, DL340 CPU Battery Replacement

The DL305 CPUs have a lithium battery to retain the application program and retentive memory when the system is not receiving power from the power supply. Typical battery life is five years. This time period includes PLC runtime and normal shutdown periods such as preventative maintenance and power outages.
The CPU has indicators which tell when it is necessary to change the battery. However, if your battery has been in your system for an extended period of time, you may wish to take added precautions to ensure that the system memory will be retained by installing a new battery when shutting the system down for a period of more than ten days.

NOTE: Before replacing your CPU battery, you should back-up your application program. This can be done by saving the program to hard/floppy disk on a personal computer or using the handheld programmer along with a cassette tape recorder. The CPU has a built-in capacitor to retain the memory for several minutes while the battery is being replaced.

WARNING: If the battery connector is not connected to the PC beard or the battery is not installed, the indicator will notnotify you of the error. Be sure the battery is in place and the connector is firmly seated before you install the CPU inte the base.


To replace the CPU battery:

1. Turn power off to the system.
2. Wait 60 seconds then remove the CPU. Do not short any connectors or components on the CPU since it may alter the program memory.
3. Unlatch and tilt the clip covering the battery.
4. Pull the two wire battery connector from the PC board and remove the battery.

WARNING: Do not attempt to recharge the battery or dispose of it by fire. The battery may explode or release hazardous materials.

To install the CPU battery:

1. Plug the (keyed) two wire battery connector on the battery into the connector on the PC board.
2. Push gently till the connector snaps closed
3. Slide the battery under the battery retaining clip till the battery is positioned in the socket.
4. Push the retaining clip down over the battery snapping the clip over the edge of the PC board.
5. Note the date the battery was changed.

## Installing the CPU

Before you complete these steps, make sure you have set the dipswitches and/or jumpers needed to support your application.


WARNING: To minimize the risk of electrical shock, personal injury, or equipment damage, always disconnect the system power before installing or removing any system component.

When inserting the CPU into the base, align the PC board with the grooves on the top and bottom of the base. Push the CPU straight into the base until it is firmly seated in the backplane connector.


## CPU Setup and System Functions

## A Few Things to Know

Even if you have years of experience using PLCs, there are a few things you need to do before you can start entering programs. This section includes some basic things, such as changing the CPU mode and connecting a programming device. Here is a list of the items that are discussed.

- Auxiliary Functions
- Connecting a Programming Device
- Changing the CPU Modes
- Clearing the CPU memory

The following paragraphs provide the setup information necessary to get the CPU ready for programming. The actual setup information depends on the type of programming device you are using. For example, the DL305 Handheld Programmer manual provides the Handheld keystrokes required to perform all of these operations. The DirectSOFT manual provides a description of the menus and keystrokes required to perform the setup procedures via DirectSOFT.

What are Auxiliary Functions?

Many CPU tasks involve the use of predefined functions. These are often called Auxiliary (AUX) Functions. The AUX Functions perform many different operations, ranging from simple operating mode changes to determining the firmware revision number.
You can access all of the AUX Functions from DirectSOFT menu options, but not from the DL305 Handheld Programmer. You can still perform some of the operations with the Handheld Programmer, but they are accomplished by using a certain series of keystrokes rather than by entering a specific AUX function.

NOTE: Neither DirectSOFT or the Handheld Programmer utilize the numbers shown for the AUX functions. These numbers have been included because many of you may already have existing software packages that can be used with these CPUs. If you do already have an existing software package, remember that any additional features (such as added I/O, CRs, etc. available with the DL340 CPU) may not be accessible.

| AUX Function and Description |  | DL330, DL330P, DL340 |  |
| :---: | :---: | :---: | :---: |
| AUX 1* - Diagnostics and PLC Modes |  | Software | HP |
| 10 | Program Syntax Check (Grammar check) | $\bigcirc$ | $\bigcirc$ |
| 11 | Compare PLC to Disk | $\bigcirc$ | $\times$ |
| 12 | PLC Operational Mode | $\bigcirc$ | $\bigcirc$ |
| 13 | Revision Number | $\bigcirc$ | $\times$ |
| AUX 3* - Clear PLC Memory |  | Software | HP |
| 31 | Ladders | $\bigcirc$ | $\bigcirc$ |
| 32 | Data Registers | $\bigcirc$ | $\times$ |
| 33 | Timer / Counter Accumulators | $\bigcirc$ | $\times$ |
| AUX 6* - Save Data from PLC |  | Software | HP |
| 61 | Ladders | $\bigcirc$ | $\bigcirc$ |
| 62 | Data Registers | $\bigcirc$ | $\times$ |
| AUX 9* - Load Data to PLC |  |  |  |
| 91 | Ladders | $\bigcirc$ | $\bigcirc$ |
| 92 | Data Registers | $\bigcirc$ | $\times$ |
| Password Operations |  |  |  |
| None | Password | $\bigcirc$ | $\bigcirc$ |

-     - Function or keystrokes available

X — Not available

Connecting the Programming Devices

You can mount the Handheld directly to the port of the CPU, or you can use a cable. The cable, part number D3-HPCBL, is approximately 4.5 feet ( 1.5 m ) in length and provides more flexibility. There are two different handheld programmers for the DL305 CPUs. The D3-HP can be used with either the DL330 or the DL340. The D3-HPP can only be used with the DL330P. The D3-HPP supports the RLLPLUS features.
If you're using a Personal Computer with the DirectSOFT programming package, a Data Communications Unit (either RS232C or RS422) must be used to interface to the DL330/DL330P CPUs. DCUs may also be used to establish a connection between the DL305 and an operator interface or a network.
The DL340 CPU provides two built-in RS232C ports which can be used to directly connect to a personal computer, operator interface or network. The DCU may also be used with the DL340 if the built-in ports are otherwise occupied.

## Programming the DL340 CPU with either the Handheld programmer or the PC



Programming the DL330 CPU with either the Handheld programmer or the PC (using a Data Communication Unit)


Changing the CPU Mode of Operation

There are two modes of operation for the DL305 CPUs:

- RUN - executes the application program and updates I/O modules
- PGM - allows program entry, does not execute the application program or update I/O modules

The CPU modes for all DL305 CPUs can be changed by using either a Handheld Programmer or DirectSOFT. The DL330 and DL330P require a Data Communications Unit when using DirectSOFT. This is discussed later in this chapter.
Since the DL340 has the possibility of being accessed through multiple ports at the same time, the Handheld Programmer and DCU have priority over the built-in RS232C ports during mode change operations. If no Handheld Programmer or DCU is online, DirectSOFT can perform mode changes through either of the built-in ports. When the Handheld Programmer or DCU is online and a mode change is attempted with DirectSOFT, the Handheld Programmer or DCU will immediately change the mode back to the original mode. This forces the CPU mode to always correspond with the keyswitch position on the Handheld Programmer.

WARNING: The CPU will automatically change modes when you connect the Handheld Programmer if the keyswitch is set for a different mode of operation. For example, if the CPU is in Run mode and the Handheld Programmer keyswitch is set to the PRG (Program) position, the CPU will automatically enter Program mode when the Handheld is connected.


The LOAD position is used for uploading a program from CPU memory to a cassette tape, or downloading a program from cassette tape to CPU memory.

## Clearing the CPU Memory

CPU Checklist

Before you enter a new program, you should always clear the CPU memory. Only a few keystrokes are required. The next few steps show how to clear the CPU memory using the handheld programmer.
Put the handheld programmer's key switch in the PRG position. Attach the handheld programmer directly to the front of the CPU making sure that the port on the back of the programmer aligns properly with the port on the CPU and the programmer's latches connect with the slots in the base power supply. Apply power to the base. LED's on the programmer will display indicating a good connection.


You can clear the memory by using the PLC/Clear PLC sub-menu from within DirectSOFT, or you can use the following Handheld Programmer keystrokes.


NOTE: This Handheld Programmer operation only clears the program memory. Any values stored in data registers are not cleared. You do have an additional menu option within DirectSOFT that allows you to clear the data registers.

Before you proceed with the I/O configuration or programming information, make sure you have:

- set the CPU dipswitches
- selected and installed the EEPROM/UVPROM (if chosen.)
- a good understanding of the various system functions needed to setup the CPU.


## Bases, <br> Expansion Bases, and I/O Configuration

In This Chapter. . . .

- Understanding I/O Numbering and Module Placement Rules
- Base Specifications and Wiring
- Using Bases for Local or Expansion I/O Systems
- Setting the Base Switches
- Example I/O Configurations
- I/O Configurations with a 5 Slot Local CPU Base - I/O Configurations with an 8 Slot Local CPU Base - I/O Configurations with a 10 Slot Local CPU Base
- Calculating the Power Budget

F3-04DA-1 module has been retired 11-29-2023. lPease consider BRX, Productivity or CLICK systems as a replacement PLC platform.

D3-05BDC 5-slot base \& D3-10BDC 10-slot base, and D3-08TR have been retired as of 03/2022. Please consider BRX, Productivity or CLICK systems as a replacement PLC platform.

D3-05B-1 5 slot base has been retired as of 09/2021. Please consider BRX, Productivity or CLICK systems as a replacement PLC platform.

Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

# Understanding I/O Numbering and Module Placement Rules 

Before you install any I/O modules or begin installing or using the bases, it is very helpful to understand how the DL305 I/O numbering and module placement restrictions can sometimes dictate how your system is put together.

DL305 I/O
Configuration History

The DL305 product family has had several enhancements over the years. Each time the product family has grown or has been enhanced, compatibility with the earlier products has been of the utmost concern. Some of these enhancements such as increasing the I/O count and supporting 16 point modules have impacted the numbering system. To help you understand our numbering scheme we have provided the following account of how the numbering system has been affected.

- When the 16 point I/O modules were introduced to the standard line of 8 point modules, the I/O numbering system was not modified to count in 16 consecutive units. This was done to maintain compatibility with the 8 point systems. This means each 16 point module uses two groups of eight consecutive numbers such as 000 through 007 and 100 through 107.
- When the I/O count was increased from the original 112 maximum to 176 maximum (for the DL330/DL330P CPU) and 184 maximum (for the DL340 CPU), most of the new I/O addresses were not set up to be consecutive with the the original $112 \mathrm{I} / \mathrm{O}$. This means you will see a large jump in the $\mathrm{I} / \mathrm{O}$ number ranges.

Octal Numbering System

The DL305 I/O points are numbered in octal (base 8.) The octal numbering system does not include the numbers 8 and 9 . The following table lists the first few octal numbers with the equivalent decimal numbers so you can see the numbering pattern.

| Octal <br> Numbers | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 20 | 21 | 22 | 23 | 24 | $\ldots$ |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| Decimal <br> Numbers | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | $\ldots$ |

Fixed I/O Numbering

The DL305 base I/O numbering is fixed, you cannot choose the I/O address of specific points since the system allocates the addresses for each slot. The I/O number ranges are 0-177 and 700-767. The I/O numbering for each slot in the base depends on two things:

1. The base configuration, which is determined by the size of the base and whether you are using an expansion base.
2. The number of $I / O$ points per module and the location of the $I / O$ modules in the base.

## I/O Numbering Guidelines

I/O numbering begins with address " 000 " which is the slot adjacent to the CPU. Each module uses increments of eight I/O points. For 8 point modules the I/O addresses are made up of eight contiguous points for each module. For 16 point modules the I/O addresses are made up of two groups of eight contiguous points, the first group follows the same scheme as the 8 point module and the second group adds 100 to the values of the first group.
The examples below show the I/O numbering for a 5 slot local CPU base with 8 point I/O and a 5 slot local CPU base with 16 point I/O.
5 Slot Base Using 8 Point I/O Modules 5 Slot Base Using 16 Point I/O Modules


Slot Number: 3-2-1-0


Slot Number: 3-2-1-0

## Number of I/O

Points Required for
Each Module

| DC Input Modules |  | DC Output Modules |  | Relay Output Modules |  | Analog Modules (cont.) |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| D3-08ND2 | 8 | D3-08TD1 | 8 | D3-08TR | 8 | F3-04DA-1 | 16 |
| D3-16ND2-1 | 16 | D3-08TD2 | 8 | F3-08TRS-1 | 8 | F3-04DA-2 | 16 |
| D3-16ND2-2 | 16 | D3-16TD1-1 | 16 | F3-08TRS-2 | 8 | F3-04DAS | 16 |
| D3-16ND2F | 16 | D3-16TD1-2 | 16 | D3-16TR | 16 | ASCII BASIC |  |
| F3-16ND3 | 16 | D3-16TD2 | 16 | Analog Modu |  | F3-AB128-R | 16 |
| AC Input Modules |  | AC Output Modules |  | D3-04AD | 16 | F3-AB128-T | 16 |
| D3-08NA-1 | 8 | D3-04TAS | 8* | F3-04ADS | 16 | F3-AB128 | 16 |
| D3-08NA-2 | 8 | F3-08TAS | 8 | F3-08AD | 16 | F3-AB64 | 16 |
| D3-16NA | 16 | D3-08TA-1 | 8 | F3-08TEMP | 16 | Specialty Mo |  |
| AC/DC Input Modules |  | D3-08TA-2 | 8 | F3-08THM-n | 16 | D3-08SIM | 8 |
| D3-08NE3 | 8 | F3-16TA-1 | 16 | F3-16AD | 16 | D3-HSC | 16 |
| D3-16NE3 | 16 | D3-16TA-2 | 16 | D3-02DA | 16 |  |  |

* This is a 4-point module, but each slot is assigned a minimum of $8 \mathrm{I} / \mathrm{O}$ points.

Retired Modules:
D3-16ND2-1, D3-16NA, D3-16TD-1, D3-16TD-2, D3-08TA-2, F3-08TAS-1, F3-04DA-1.

## I/O Module Placement Rules

There are some limitations that determine where you can place certain types of modules. Some modules require certain locations and may limit the number or placement of other modules. We have tried to give clearly written explanations of the rules governing module placement, but we realize a picture can sometimes be worth a thousand words. If you have difficulty with some of our explanations, please look ahead to the illustrations in this chapter. They should clear up any gray areas in the explanation and you will probably find the configuration you intend to use in your installation.
In all of the configurations mentioned the number of slots from the CPU that are to be used can roll over into an expansion base if necessary. For example if a rule states a module must reside in one of the six slots adjacent to the CPU, and the system configuration is comprised of two 5 slot bases, slots 1 and 2 of the expansion base are valid locations.
The following table provides the general placement rules for the DL305 components.

| Module | Restriction |
| :--- | :--- |
| CPU | The CPU must reside in the first slot of the local CPU <br> base. The first slot is the closest slot to the power supply. |
| 16 Point I/O <br> Modules | There can be a maximum of eight 16 point modules <br> installed in a system depending on the CPU type and I/O <br> modules used. The 16 point modules must be in the first 8 <br> slots adjacent to the CPU rolling over into an expansion <br> base if necessary. If any of the eight slots adjacent to the <br> CPU are not used for 16 point modules, they can be used <br> for 8 point modules. |
| Analog Modules | Analog modules must reside in any valid 16 point I/O slot. |
| ASCII Basic <br> Modules | ASCII Basic modules must reside in any valid 16 point I/O <br> slot. |
| High Speed <br> Counter | High Speed Counters may be used in one of the first 4 <br> slots in the local CPU base. |

DL330/DL330P Rules for 16 Point Modules

The DL330 CPU can address up to seven 16 point modules as long as they reside in the seven slots adjacent to the CPU, however; there is one circumstance where the number of 16 point modules can be limited.

- Only six 16 point modules can be used if High Speed Counter modules are installed in the system. The 16 point modules must reside in the six slots adjacent to the CPU.

NOTE: The High Speed Counter module is considered to be a 16 point module.


Addresses 160-167 are not available as I/O if High Speed Counter modules are used in the system

NOTE: Addresses 160-167 are normally used as CRs, but they can also be used as I/O for 16 point modules. You cannot use the points as both CRs and I/O. Also, when you use these as I/O points, you still enter them as C160-C167 in DirectSOFT.

DL340 Rules for 16 Point Modules

The DL340 CPU can address up to eight 16 point modules as long as they reside in the eight slots adjacent to the CPU, however; there are circumstances where the number of 16 point modules can be limited.

1. Only seven 16 point modules can be used if a Thumbwheel Interface module is installed in the system. The 16 point modules must reside in the seven slots adjacent to the CPU.
2. Only seven 16 point modules can be used if High Speed Counter modules are installed in the system. The 16 point modules must reside in the six slots adjacent to the CPU, skipping one slot, and using the 8th slot from the CPU for the last of the 16 point modules.
3. Only six 16 point modules can be used if a High Speed Counter and a Thumbwheel Interface module are installed in the system. The 16 point modules must reside in in the six slots adjacent to the CPU .

NOTE: Both High Speed Counter and Thumbwheel Interface modules are considered to be 16 point modules.


- Addresses 170-177 are not available as I/O if a Thumbwheel Interface module is used in the system
- Addresses 160-167 are not available as I/O if High Speed Counter modules are used in the system
: Addresses 160-167 and 170-177 are not available as I/O if both High Speed Counters and a Thumbwheel interface module are used in the system.

NOTE: Addresses 160-177 are normally used as CRs, but they can also be used as I/O points if you are using 16 point modules. Remember, if you use these locations as I/O points you cannot use them as CRs. Also, when you use these as I/O points, you still enter them as C160-C177 in DirectSOFT.

## Base Specifications and Wiring

Three Sizes of Bases

There are three base sizes available to hold your I/O modules: 5, 8 and 10 slot. The 5 and 10 slot bases can be used as either a local CPU base or an expansion base. The 8 slot base can only be used as a local CPU base. The 5, 8, and 10 slot bases are available with a built-in 110/220 VAC power supply. The 5 slot base is also available with a built-in 24 VDC power supply.
Remote I/O is not offered in the DL305 product family. All DL305 products, with the exception of the DL340 CPU, are compatible with remote I/O systems previously offered by GE FANUC® and TEXAS INSTRUMENTS®


Bases and Maximum I/O Supported

## Base Mounting Dimensions

The maximum I/O for the base combinations is shown below. The number of I/O points supported also depends on the which CPU is used in the system.

| Base Configuration | DL330 / DL330P <br> CPU | DL340 CPU |
| :--- | :--- | :--- |
| 5 slot local CPU base system | 64 I/O max. | 64 I/O max |
| 5 slot local CPU base system with a 5 slot <br> expansion base | 120 I/O max. | 128 I/O max. |
| 5 slot local CPU base system with two 5 <br> slot expansion bases | 120 I/O max. | 128 I/O max. |
| 8 slot local CPU base system | 112 I/O max. | 112 I/O max. |
| 8 slot local CPU base system with a 5 slot <br> expansion base | 152 I/O max. | 152 I/O max. |
| 10 slot local CPU base system | 128 I/O max. | 136 I/O max. |
| 10 slot local CPU base system with a 5 <br> slot expansion base | 168 I/O max. | 176 I/O max. |
| 10 slot local CPU base system with a 10 <br> slot expansion base | 176 I/O max. | 184 I/O max. |

Use these mounting dimensions when you install the DL305 bases. Make sure you have followed the installation guidelines shown in Chapter 2 for proper spacing.


Connecting the Power Supply


The following diagram shows the terminal connections located on the power supply of the DL305 bases.

WARNING: Damage will occur to the base power supply if 230 VAC is connected to the 115 VAC terminal connections. Once the power wiring is connected, install the protective cover. When the cover is removed there is a risk of electrical shock if you accidentally touch the connection terminals.


24 VDC
Terminal Strip


Expansion Base Power Supply Wiring Example

The following diagram shows how to connect the power when you use both local CPU and Expansion bases.


## Base <br> Specifications

|  | D3-05B | D3-05BDC | D3-08B | D3-10B |
| :---: | :---: | :---: | :---: | :---: |
| Number of Slots | 5 | 5 | 8 | 10 |
| Local CPU Base | Yes | Yes | Yes | Yes |
| Expansion Base | Yes | Yes | No | Yes |
| Input Voltage Range | $\begin{aligned} & \text { 97-132 VAC } \\ & 194-264 \text { VAC } \\ & 47-63 \mathrm{~Hz} \end{aligned}$ | 20.5-30 VDC <10\% ripple | $\begin{aligned} & 97-132 \text { VAC } \\ & 194-264 \text { VAC } \\ & 47-63 \mathrm{~Hz} \end{aligned}$ | $\begin{aligned} & 97-132 \text { VAC } \\ & 194-264 \text { VAC } \\ & 47-63 \mathrm{~Hz} \end{aligned}$ |
| Base Power Consumption | 70 VA max (46W) | 48 Watts | 70 VA max (57W) | 70 VA max (57W) |
| Inrush Current max. | 30A | 30A | 30A | 30A |
| Dielectric Strength | 1500VAC for 1 minute between terminals of AC P/S, Run output, Common, 24VDC | 1500VAC for 1 minute between 24VDC input terminals and Run output | 1500VAC for 1 minute between terminals of AC P/S, Run output, Common, 24VDC | 2000VAC for 1 minute between terminals of AC P/S, Run output, Common, 24VDC |
| Insulation Resistance | $>10 \mathrm{M} \Omega$ at 500VDC | $>10 \mathrm{M} \Omega$ at 500VDC | $>10 \mathrm{M} \Omega$ at 500VDC | $>10 \mathrm{M} \Omega$ at 500VDC |
| Power Supply Output (Voltage Ranges and Ripple) | (5VDC) 4.75-5.25V less than $0.1 \mathrm{~V} p-\mathrm{p}$ (9VDC) 8.5-13.5V less than 0.2 V p-p (24VDC) 20-28V less than 1.2 V p-p | (5VDC) 4.75-5.25V less than $0.1 \mathrm{~V} p-\mathrm{p}$ (9VDC) 8.5-13.5V less than $0.2 \mathrm{~V} p-\mathrm{p}$ (24VDC) 20-28V less than $1.2 \mathrm{~V} p-\mathrm{p}$ | (5VDC) 4.75-5.25V less than 0.1 V p-p (9VDC) 8.0-12.0V less than $0.2 \mathrm{~V}-\mathrm{p}$ (24VDC) 20-28V less than 1.2 V p-p | (5VDC) 4.75-5.25V less than 0.1 V p-p (9VDC) 8.0-12.0V less than 0.2 V p-p (24VDC) 20-28V less than 1.2 V p-p |
| 5 VDC current available | 1.4A * | 1.4A | $\begin{aligned} & 1.4 \mathrm{~A} @ 122^{\circ} \mathrm{F}\left(50^{\circ} \mathrm{C}\right) \\ & 1.0 \mathrm{~A} @ 140^{\circ} \mathrm{F}\left(60^{\circ} \mathrm{C}\right) \end{aligned}$ | $\begin{aligned} & 1.4 \mathrm{~A} @ 122^{\circ} \mathrm{F}\left(50^{\circ} \mathrm{C}\right) \\ & 1.0 \mathrm{~A} @ 140^{\circ} \mathrm{F}\left(60^{\circ} \mathrm{C}\right) \end{aligned}$ |
| 9 VDC current available | 0.8A * | 0.8A | $\begin{aligned} & 1.7 \mathrm{~A} @ 122^{\circ} \mathrm{F}\left(50^{\circ} \mathrm{C}\right) \\ & 1.4 \mathrm{~A} @ 140^{\circ} \mathrm{F}\left(60^{\circ} \mathrm{C}\right) \end{aligned}$ | $\begin{aligned} & \hline 1.7 \mathrm{~A} @ 122^{\circ} \mathrm{F}\left(50^{\circ} \mathrm{C}\right) \\ & 1.4 \mathrm{~A} @ 140^{\circ} \mathrm{F}\left(60^{\circ} \mathrm{C}\right) \end{aligned}$ |
| 24 VDC current available | 0.5A * | 0.5A | 0.6A | 0.6A |
| Auxiliary 24 VDC Output | 100mA max | None | 100mA max | 100mA max |
| Run Relay | 250 VAC, <br> 4A (resistive load) | $\begin{aligned} & 250 \text { VAC, } \\ & \text { 4A (resistive load) } \end{aligned}$ | 250 VAC, <br> 4A (resistive load) | 250 VAC, <br> 4A (resistive load) |
| Fuses | 2A (250V) <br> User replaceable | 4A (250V) <br> User replaceable | 2A (250V) <br> User replaceable | 2A (250V) <br> User replaceable |
| Dimensions WxHxD | $11.42 \times 4.85 \times 4.41 \mathrm{in}$. (290×123×112 mm) | $\begin{aligned} & 11.42 \times 4.85 \times 4.41 \mathrm{in} \\ & (290 \times 123 \times 112 \mathrm{~mm}) \end{aligned}$ | $15.55 \times 4.85 \times 4.41$ in ( $395 \times 123 \times 112 \mathrm{~mm}$ ) | $18.3 \times 4.85 \times 4.41 \mathrm{in}$. ( $465 \times 123 \times 112 \mathrm{~mm}$ ) |
| Weight | 34 oz. (964g) | 34 oz. (964g) | 44.2 oz. (1253g) | 50.5 oz. (1432g) |

* The total current for the D3-05B must not exceed 2.3A.

Auxiliary 24VDC Output at Base Terminal

There is 24 VDC available from the 24 VDC output terminals on all bases except the 5 slot DC version (D3-05BDC). The 24 VDC supply can be used to power external devices or DL305 modules that require external 24 VDC. The power used from the this 24 VDC output reduces the internal system 24 VDC available to the modules by an equal amount. So if you use this power supply, make sure you consider this when you calculate the power budget. (The power budget is discussed in more detail later in this chapter.)

Power Supply Schematics

The following diagram shows the details of how the DL305 base provides many of the specifications listed on the previous page.

Schematic for D3-05B, D3-08B, D3-10B


Schematic for D3-05BDC


Using the Run
Relay on the Base Power Supply

The RUN relay output, located on the DL305 base power supply, can be used to detect an undesired failure on the local CPU base or an expansion base. The following table shows the operating characteristics of the RUN relay for a local CPU base or an expansion base.

| Event | Local CPU Base <br> RUN Relay Would: | Expansion Base <br> RUN Relay Would: |
| :--- | :--- | :--- |
| PROGRAM to RUN mode <br> Transition | Energize | Not change |
| The CPU detects a fatal <br> error | De-energize | Not change |
| CPU Local Base is <br> Removed Form the RUN <br> Mode | De-energize | Not change |
| Power Source to the <br> Power Supply is Turned <br> OFF | De-energize | De-energize |
| 9 VDC or 24 VDC Failure <br> on the Power Supply | De-energize | De-energize |

The following example demonstrates possible uses for the RUN relay on the DL305 bases.


Use of the RUN relay to shutdown critical field devices upon error detection

Use of the RUN relay to monitor system operation

Installing CPUs and I/O Modules

The CPU must go into first slot (next to the power supply) on the far right side of the base. When inserting components into the base, align the PC board(s) of the module with the grooves on the top and bottom of the base. Push the module straight into the base until it is firmly seated in the backplane connector. To remove a module from the base squeeze the tabs on the top and bottom of the faceplate and pull the module straight out.


WARNING: Do not remove any system component when system power is on. This may cause damage to the system or unpredictable system operation that can result in a risk of personal injury.

## Using Bases for Local or Expansion I/O Systems

Base Uses Table

It is helpful to understand how you can use the various DL305 bases in your control system. The following table shows how the bases can be used.

| Base Part \# | Number of Slots | Can Be Used As <br> A Local CPU <br> Base | Can Be Used As <br> An Expansion <br> Base |
| :--- | :--- | :--- | :--- |
| D3-05B | 5 | Yes | Yes |
| D3-05BDC | 5 | Yes | Yes |
| D3-08B | 8 | Yes | No |
| D3-10B | 10 | Yes | Yes |

Local/Expansion Connectivity

The configurations below show the valid combinations of local CPU bases and expansion bases.

NOTE: You should use one of the configurations listed below when designing an expansion system. If you use a configuration not listed below the system will not function properly.

5 slot local CPU base with a maximum of two 5 slot expansion bases

8 slot local CPU base with a 5 slot expansion base


10 slot local CPU base with a 5 slot expansion base


10 slot local CPU base with a 10 slot expansion base


## Connecting Expansion Bases

The local CPU base is connected to the expansion base using a 1.5 ft . cable (D3-EXCBL). The base must be connected as shown in the diagram below.

The top expansion connector on the base is the input from a previous base. The bottom expansion connector on the base is the output to an expansion base. The expansion cable is marked with "CPU Side" and "Expansion Side". The" CPU Side" of the cable is connected to the bottom port of the base and the "Expansion Side" of the cable is connected to the top port of the next base.


Note: Avoid placing the expansion cable in the same wiring tray as the I/O and power source wiring.

## Setting the Base Switches

5 Slot Bases
The 5 slot and 10 slot bases have jumper switches that need to be set depending on which system configuration is used. The 8 slot base does not have any switches.
The 5 slot bases have a two position toggle switch which is used to set the base as the CPU local base, the first expansion base, or the second (last) expansion base.
The switch is set to the " 1,3 " position if the base is the local CPU base or the third base in the system. The switch is set to the " 2 " position if the base is the 2nd base in the system. If the 5 slot base is used as an expansion base for a 10 slot local CPU base the switch is set in the " 1,3 " position.

See Errata Sheet at the beginning of this file. Newer 5 slot bases have a jumper switch instead of a toggle switch to set the base as the CPU local base or as an expansion base.


10 Slot Base
The 10 slot base has a jumper switch between slot 3 and 4 used to set the base to local CPU base or expansion base. There is also a jumper switch between slot 9 and 10 that sets slot 10 to the 100-107 I/O address range or to the 700-707 I/O address range.


## Example I/O Configurations

The following system configurations will allow you to quickly configure your system by using examples. These system configurations show the I/O numbering and the base switch settings for every valid base configuration for a DL305 system.
16 Point I/O
Allocation Example
When a 16 point I/O module is used the last 8 I/O addresses of each 16 point module could have been used in another base slot. In the illustration below Example A shows a 16 point module in the slot next to the CPU using address 000-007 and 100-107. The expansion I/O cannot use the last slot of the expansion base since it is assigned addresses 100-107 and the 16 point module next to the CPU has already used these addresses. Example B shows an 8 point module in the slot next to the CPU and an 8 point module in the last slot of the expansion base. Both examples are valid configurations.

Example A

## See Errata Sheet at

 the beginning of this file. Newer 5 slot bases have a jumper switch instead of a toggle switch to set the base as the CPU local base or as an expansion base.Example B


Local CPU Base

## Examples Show Maximum I/O Points Available

For the following examples the configurations using 16 point I/O modules are shown with the maximum I/O points supported so you can always reduce the I/O count in one of our examples and the configuration will still be valid. Substitution of 8 point I/O modules can be made in place of any of the 16 point modules without affecting the I/O numbering for any of the other I/O modules. When a 16 point module is replaced with a 8 point I/O module the last 8 I/O addresses of that 16 point module may or may not be useable in another slot location, depending on the system configuration used

## I/O Configurations with a 5 Slot Local CPU Base

The configurations below and on the next few pages show a 5 slot base with 8 point I/O modules, 16 point modules, one expansion base and two expansion bases.

Switch settings

5 Slot Base with 8 Point I/O

The 5 slot base has a toggle switch on the inside of the base between slots 4 and 5 which allows you to select:

| Type of Base | Switch Position |
| :--- | :--- |
| Local CPU | Base 1,3 |
| First Expansion | Base 2* |
| Last Expansion | Base 1,3 |

*used only with a 5 slot local CPU base

See Errata Sheet at the beginning of this file. Newer 5 slot bases have a jumper switch instead of a toggle switch to set the base as the CPU local base or as an expansion base.

5 Slot Base and 5 Slot Expansion 5 Slot Expansion
Base with 8 Point I/O

See Errata Sheet at the beginning of this file. Newer 5 slot bases have a jumper switch instead of a toggle switch to set the base as the CPU local base or as an

Total I/O: 72 expansion base.

| See Errata Sheet at |
| :--- |
| the beginning of this |
| file. Newer 5 slot |
| bases have a jumper |
| switch instead of a |
| toggle switch to set the |
| base as the CPU local |
| base or as an |
| expansion base. |



5 Slot Base and 5 Slot Expansion Base with 16 Point I/O

Total I/O: 128


NOTE: If a 16 pt module is used in the last two available slots of the expansion base, 160 through 177 will not be available for control relay assignments. Also, even though you are using these points as I/O, you still enter them as $\mathrm{C} 160-\mathrm{C} 177$ in DirectSOFT.

5 Slot Base and
Two 5 Slot
Expansion Bases with 8 Point I/O

Total I/O: 112

See Errata Sheet at the beginning of this file. Newer 5 slot bases have a jumper switch instead of a toggle switch to set the base as the CPU local base or as an expansion base.


5 Slot Base and Two 5 Slot Expansion Bases with 16 and 8 Point I/O

Total I/O: 128


NOTE: If a 16 pt module is used in the last two available slots of the expansion base, 160 through 177 will not be available for control relay assignments. Also, even though you are using these points as I/O, you still enter them as C160-C177 in DirectSOFT.

## I/O Configurations with an 8 Slot Local CPU Base

The configurations below show an 8 slot base with 8 point I/O modules, 16 point modules, one 5 slot expansion base and two 5 slot expansion bases.

8 Slot Base with 8 Point I/O

8 Slot Base with 16 Point I/O

Total I/O: 56


Total I/O: 112
*See note below regarding points 160-167


Total I/O: 96
beginning of this file. Newer 5 slot bases have a jumper switch instead of a toggle switch to set the base as the CPU local base or as an expansion base.

8 Slot Base and 5 Slot Expansion Base with 16 Point I/O

8 Slot Base and 5 Slot Expansion Base with 8 Point I/O

See Errata Sheet at the


Total I/O: 152
*See note below regarding points 160-167


NOTE: If a 16pt module is used in Slot 6, 160 through 167 will not be available for control relay assignments. Also, even though you are using these points as I/O, you still enter them as C160-C167 in DirectSOFT.

## I/O Configurations with a 10 Slot Local CPU Base

## Switch settings



Last Slot Address Range 100 to 107

Last Slot Address Range 700 to 707

The configurations below and on the next few pages show a 10 slot base with 8 point I/O modules, with 16 point modules, with a 5 slot expansion base and with a 10 slot expansion base.
The 10 slot base has two jumper switches to select the base type and the address ranges to use. These switches can be found on the base between slots 3 and 4 (SW1) and slots 9 and 10 (SW2). Jumper switch SW1 is used to select if the base is a local CPU base or an expansion base. Jumper switch SW2 determines the I/O address range (100-107 or 700-707) for the 10th slot on the local CPU base. By selecting the address range of 700 to 707 for slot 10 , it is possible to use a 16 point module next to the CPU (which uses the ranges of 000 to 007 and 100 to 107), however; the position of this switch will affect the I/O numbering for the expansion I/O if used.

NOTE: Jumper switch SW2 must be set to "100 EXP" on the expansion base.

## Total I/O: 72



## Total I/O: 72



Jumper SW1


## 10 Slot Expansion Base with 16 Point I/O

Configuration 1

The next two configurations show a local CPU base using 16 point I/O modules and the two possibilities for how to configure the base to use the maximum number of $\mathrm{I} / \mathrm{O}$ points.

Configuration 1 shows an 8 point I/O module the slot next to the CPU and the address range of 100-107 for the last slot. When jumper switch SW2 is set to the "100 EXP" position, the address range for the last slot is set to 100-107, thereby limiting the address range for the first module to 000-007. This means if you use this configuration, the first module must be an 8 point I/O module. You will have more available addresses for an expansion base as you will see in the example using a 10 slot expansion base.

Total I/O:128
Configuration 1

*See note below regarding points 160-167 and 170-177.

Configuration 2
*See note below regarding points 160-167 and 170-177.

Total I/O: 136
Configuration 2

Configuration 2 shows a 16 point I/O module in the slot next to the CPU and the address range of 700-707 for the last slot. This is the maximum I/O configuration for a 10 slot local CPU base. When jumper switch SW2 is set to the " 700 " position the address range for the last slot is set to 700-707 making addresses 000-007 and 100-107 available for use in the first slot. The position of jumper switch SW2 can limit the amount of I/O addresses available to the larger expansion bases since expansion I/O numbering would normally start with address 700 .


NOTE: If a 16pt module is used in Slot 6 for the DL330 or DL330P CPU, 160 through 167 will not be available for control relay assignments. If a 16pt module is used in Slot 6 and/or Slot 7 for a DL340 CPU, 160-167 and/or 170-177 are not available for control relay assignments. Also, even though you are using these points as I/O, you still enter them as C160-C167/C170-C177 in DirectSOFT.

10 Slot Base and 5 Slot Expansion Base with 16 Point I/O

Total I/O: 176


NOTE: If a 16 pt module is used in Slot 6 for the DL330 or DL330P CPU, 160 through 167 will not be available for control relay assignments. If a 16 pt module is used in Slot 6 and/or Slot 7 for a DL340 CPU, 160-167 and/or 170-177 are not available for control relay assignments. Also, even though you are using these points as $\mathrm{I} / \mathrm{O}$, you still enter them as C160-C167/C170-C177 in DirectSOFT.

Expansion Addresses Depend on Local CPU Base Configuration．

I／O addresses change depending on the point configuration in the local CPU base． Notice，when the local CPU base contains only 8 point I／O modules，addresses 110－117，120－127 and 130－137 are available for use in the expansion base．When the local CPU base has 16 point I／O modules，which use the I／O addresses 110－117， 120－127 and 130－137，these addresses are taken up and are not available for use in the expansion base．

10 Slot Base and 10 Slot Expansion
Base with 8 Point I／O

10 Slot Base and 10 Slot Expansion Base with 16 Point I／O

Total I／O： 152

Jumper SW2


SW2



Jumper SW1


SW1

| EXP |  | CPU |
| :---: | :---: | :---: |
|  |  |  |
|  | 吅 | $\square$ |
| 品 | － | － |
| 0 | － | － |
| 0 | 吅 | － |
| － | － | － |
| 回 | 吅 | $\cdots$ |
|  |  |  |

Total I／O： 184
Jumper
SW2


NOTE：If a 16pt module is used in Slot 6 for the DL330 or DL330P CPU， 160 through 167 will not be available for control relay assignments．If a 16pt module is used in Slot 6 and／or Slot 7 for a DL340 CPU，160－167 and／or 170－177 are not available for control relay assignments．Also，even though you are using these points as I／O，you still enter them as C160－C167／C170－C177 in DirectSOFT．

## Calculating the Power Budget

Managing your
Power Resource


## Auxiliary Base <br> Power Source

## Base Power Specifications

When you determine the types and quantity of I/O modules you will be using in the DL305 system it is important to remember there is a limited amount of power available from the power supply. We have provided a chart to help you easily see the amount of power available with each base. The following chart will help you calculate the amount of power you need with your I/O selections. At the end of this section you will also find an example of power budgeting and a worksheet for your own calculations.
If the I/O you choose exceeds the maximum power available from the power supply you can resolve the problem in one of two ways:

- Shift some of the modules to an expansion base which contains another power supply.
- If a 5 slot base is being used, replace it with an 8 or 10 slot base. This will provide more power on the 9 V and 24 V power supplies.


#### Abstract

WARNING: It is extremely important to calculate the power budget. If you exceed the power budget, the system may operate in an unpredictable manner which may result in a risk of personal injury or equipment damage.


There is 24 VDC available from the 24 VDC output terminals on the bases (except D3-05BDC). The 24 VDC can be used to power external devices or DL305 modules that require external 24 VDC. The power used from this supply reduces the internal system 24 VDC available to the modules by an equal amount. When using the 24 VDC output at the base terminal it is not recommended to exceed 100 mA .
This chart shows the amount of current available for the three voltages supplied on DL305 bases. Use these currents when calculating the power budget for your system.

| Bases | 5V Power <br> Supplied in <br> mA | 9V Power <br> Supplied in <br> mA | 24V Power <br> Supplied in <br> mA | Auxiliary <br> 24 VDC <br> Output at <br> Base Terminal |
| :--- | :--- | :--- | :--- | :--- |
| D3-05B | 1400 | 800 | 500 | Yes |
| D3-05BDC | 1400 | 800 | 500 | None |
| D3-08B | 1400 | 1700 | 600 | Yes |
| D3-10B | 1400 | 1700 | 600 | Yes |

NOTE: The total current for the D3-05B and D3-05BDC should not exceed 2.3 Amps. The base currents listed for the D3-08B and the D3-10B are for operating ambient temperatures between $0^{\circ} \mathrm{C}$ and $50^{\circ} \mathrm{C}$.

Module Power Requirements

The next three pages show the amount of maximum current required for each of the DL305 modules. The column labeled "External Power Source Required" is for module operation and is not for field wiring. Use these currents when calculating the power budget for your system. If 24 VDC is needed for external devices, the 24 VDC ( 100 mA maximum) output at the base terminal strip may be used as long as the power budget is not exceeded.

|  | 5V Power Required in mA | 9V Power Required in mA | 24V Power Required in mA | External Power Source Required |
| :---: | :---: | :---: | :---: | :---: |
| CPUs |  |  |  |  |
| D3-330 | 300 | 50 | 0 | None |
| D3-330P | 300 | 50 | 0 | None |
| D3-340 | 300 | 20 | 0 | None |
| Specialty CPUs |  |  |  |  |
| F3-OMUX-1 | 300 | 0 | 0 | None |
| F3-OMUX-2 | 300 | 0 | 150 | None |
| F3-PMUX | 500 | 0 | 0 | None |
| F3-RTU | 300 | 0 | 0 | 0 |
| DC Input Modules |  |  |  |  |
| D3-08ND2 | 0 | 10 | 112 | None |
| D3-16ND2-1 | 0 | 25 | 224 | None |
| D3-16ND2-2 | 0 | 24 | 209 | None |
| D3-16ND2F | 0 | 25 | 224 | None |
| F3-16ND3F | 0 | 148 | 68 | None |
| AC Input Modules |  |  |  |  |
| D3-08NA-1 | 0 | 10 | 0 | None |
| D3-08NA-2 | 0 | 10 | 0 | None |
| D3-16NA | 0 | 100 | 0 | None |
| AC/DC Input Modules |  |  |  |  |
| D3-08NE3 | 0 | 10 | 0 | None |
| D3-16NE3 | 0 | 130 | 0 | None |

Module Power Requirements (continued)

|  | 5V Power Required in mA | 9V Power Required in mA | 24V Power Required in mA | External <br> Power Source <br> Required |
| :---: | :---: | :---: | :---: | :---: |
| DC Output Modules |  |  |  |  |
| D3-08TD1 | 0 | 20 | 24 | None |
| D3-08TD2 | 0 | 30 | 0 | None |
| D3-16TD1-1 | 0 | 40 | 96 | None |
| D3-16TD1-2 | 0 | 40 | 96 | None |
| D3-16TD2 | 0 | 180 | 0 | None |
| AC Output Modules |  |  |  |  |
| D3-04TAS | 0 | 12 | 0 | None |
| F3-08TAS | 0 | 80 | 0 | None |
| F3-08TAS-1 | 0 | 25 | 0 | None |
| D3-08TA-1 | 0 | 96 | 0 | None |
| D3-08TA-2 | 0 | 160 | 0 | None |
| F3-16TA-2 | 0 | 250 | 0 | None |
| D3-16TA-2 | 0 | 400 | 0 | None |
| Relay Output Modules |  |  |  |  |
| D3-08TR | 0 | 360 | 0 | None |
| F3-08TRS-1 | 0 | 296 | 0 | None |
| F3-08TRS-2 | 0 | 296 | 0 | None |
| D3-16TR | 0 | 480 | 0 | None |
| Analog |  |  |  |  |
| D3-04AD | 0 | 55 | 0 | $\begin{aligned} & \text { 24VDC @ } \\ & 65 \mathrm{~mA} \text { max } \end{aligned}$ |
| F3-04ADS | 0 | 183 | 50 | None |
| F3-08AD | 0 | 25 | 37 | None |
| F3-08TEMP | 0 | 25 | 37 | None |
| F3-08THM-n | 0 | 50 | 34 | None |
| F3-16AD | 0 | 33 | 47 | None |
| D3-02DA | 0 | 80 | 0 | 24VDC @ 170mA max |
| F3-04DA-1 | 0 | 144 | 108 | None |
| F3-04DA-2 | 0 | 144 | 108 | None |
| F3-04DAS | 0 | 154 | 145 | None |


|  | 5V Power <br> Required in <br> mA | 9V Power <br> Required in <br> mA | 24V Power <br> Required in <br> mA | External <br> Power Source <br> Required |
| :--- | :--- | :--- | :--- | :--- |
| Communications <br> and Networking |  |  |  |  |
| D3-232-DCU | 500 | 0 | 0 | Optional 5VDC <br> @ 500mA |
| D3-422-DCU | 500 | 0 | Optional 5VDC <br> @ 500mA |  |
| F3-UNICON | 0 | 0 | (24 VDC or <br> 5 VDC) @ <br> 100mA |  |
| ASCII BASIC <br> Modules |  |  | 0 |  |
| F3-AB128-R | 0 | 205 | 0 | None |
| F3-AB128-T | 0 | 205 | 0 | None |
| F3-AB128 | 0 | 90 | 0 | None |
| F3-AB64 | 0 | 90 | 0 | None |
| Specialty <br> Modules |  | 50 | 0 | None |
| D3-08SIM | 0 | 10 | 0 | Optional 5VDC |
| D3-HSC | 0 | 50 | 0 | Optional |
| D3-PWU | 800 | 50 | 0 | 0 |
| Programming |  | 0 | 0 |  |
| D3-HP |  |  | 0 |  |
| D3-HPP |  |  | 0 |  |

Power Budget Calculation Example

The following example shows how to calculate the power budget for the DL305 system.
$\left.\begin{array}{|l|l|l|l|l|l|}\hline \text { Base \# } & & & & \begin{array}{l}\text { 24 VDC (mA) and/or } \\ \text { Auxiliary Base } \\ \text { Power Source }\end{array} \\ \text { 24 VDC Output (mA) }\end{array}\right]$

1. Using the tables at the beginning of the Power Budgeting section of this chapter fill in the information for the Base, CPU, I/O modules, and any other devices that will use system power including devices that use the 24 VDC output. Pay special attention to the current supplied by the base which you have selected since they do differ. Devices which fall into the "Other" category are devices such as the Data Communications Unit and the Handheld programmer which plug onto the CPU.
2. Add the current columns starting with slot 1 and put the total in the row labeled "Maximum power required".
3. Subtract the row labeled "Maximum power required" from the row labeled "Base Used". Place the difference in the row labeled "Remaining Power Available".
4. If "Maximum Power Required" is greater than "Base Used" in any of the three columns, the power budget will be exceeded. It will be unsafe to use this configuration and you will need to restructure your base/module configuration.

Power Budget
Calculation
Worksheet

This blank chart is provided for you to copy and use in your power budget calculations.

| Base \# | Module Type | 5 VDC (mA) | 9 VDC (mA) | 24 VDC (mA) and/or <br> Auxiliary Base <br> Power Source <br> 24 VDC Output (mA) |
| :--- | :--- | :--- | :--- | :--- |
|  |  |  |  |  |
| Base Used |  |  |  |  |
| Slot 1 |  |  |  |  |
| Slot 2 |  |  |  |  |
| Slot 3 |  |  |  |  |
| Slot 4 |  |  |  |  |
| Slot 5 |  |  |  |  |
| Slot 6 |  |  |  |  |
| Slot 7 |  |  |  |  |
| Slot 8 |  |  |  |  |
| Slot 9 |  |  |  |  |
| Slot 10 |  |  |  |  |
|  |  |  |  |  |
| Other |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Maximum power required |  |  |  |  |
| Remaining Power Available |  |  |  |  |

1. Using the tables at the beginning of the Power Budgeting section of this chapter fill in the information for the Base, CPU, I/O modules, and any other devices that will use system power including devices that use the 24 VDC output. Pay special attention to the current supplied by the base which you have selected since they do differ. Devices which fall into the "Other" category are devices such as the Data Communications Unit and the Handheld programmer which plug onto the CPU.
2. Add the current columns starting with slot 1 and put the total in the row labeled "Maximum power required".
3. Subtract the row labeled "Maximum power required" from the row labeled "Base Used". Place the difference in the row labeled "Remaining Power Available".
4. If "Maximum Power Required" is greater than "Base Used" in any of the three columns, the power budget will be exceeded. It will be unsafe to use this configuration and you will need to restructure your base/module configuration.

# I/O Module Selection \& Wiring Guidelines 

In This Chapter. . . .

- I/O Selection Considerations
- Sinking and Sourcing Circuits
- DL305 Input Module Configuration Chart
- DL305 Output Module Configuration Chart
- Configuration \#1 DL305 DC Current Sourcing Input Module
- Configuration \#2 DL305 DC Current Sinking/Sourcing Input Module
- Configuration \#3 DL305 DC Current Sinking Input Module
- Configuration \#4 DL305 AC/DC Input Module
- Configuration \#5 DL305 AC Input Module
- Configuration \#6 DL305 DC Current Sinking Output Module
- Configuration \#7 DL305 DC Current Sourcing Output Module
- Configuration \#8 DL305 AC/DC Current Sink/Source (Relay) Output Module
- Configuration \#9 DL305 AC Output Module
— Solid State Field Device Wiring to DC Input Modules
- Derating Characteristics
- I/O Wiring Guidelines
- Fuse Protection


## I/O Selection Considerations

I/O Module Selection

The DL305 product family offers various types of I/O modules for interfacing many different field devices to the PLC system. There are several electrical characteristics that should be considered when choosing the proper I/O module for a field device or for obtaining required system performance. Electrical characteristics for discrete input modules and discrete output modules are discussed in Chapters 6 and 7. The DL305 family also offers several specialized modules such as analog, ASCII BASIC modules, network interface modules, high speed counter modules, etc. These modules have their own manuals, so if you are using them you should supplement this manual with the manual specifically designed for the special module.

## Sinking and Sourcing Circuits

The charts on the following page supply information on the current sinking and current sourcing configurations using DL305 discrete I/O modules. If you have a question about the type of device required to connect to a particular module please refer to the following charts. The charts show nine common input and output module configurations. Match the module part number you are considering to the applicable configuration(s) to ensure the module type will work in your application.
For additional clarification we have included nine diagrams depicting the configurations listed in the charts. These diagrams show the module category, type of device and how they are connected to each other. The diagrams and two examples of wiring a solid state switch to an input module follow the charts on the next page.

## DL305 Input Module Configuration Chart

$\left.\left.\begin{array}{|l|c|l|l|l|l|}\hline \begin{array}{l}\text { DL305 } \\ \text { Input Module } \\ \text { Type }\end{array} & \begin{array}{l}\text { Config \#1 } \\ \text { DC Current } \\ \text { Sourcing Input }\end{array} & \begin{array}{l}\text { Config \#2 } \\ \text { DC Current } \\ \text { Sink/Source } \\ \text { Input }\end{array} & \begin{array}{l}\text { Config \#3 } \\ \text { DC Current } \\ \text { Sinking Input }\end{array}\end{array} \begin{array}{l}\text { Config \#4 } \\ \text { AC/DC Input }\end{array}\right] \begin{array}{l}\text { Config \#5 } \\ \text { AC Input }\end{array}\right]$

## DL305 Output Module Configuration Chart

| DL305 <br> Output Module Type | Config \#6 DC Current Sinking Output | Config \#7 DC Current Sourcing Output | Config \#8 AC/DC Current Sink/Source Output | Config \#9 AC Output |
| :---: | :---: | :---: | :---: | :---: |
| D3-04TD1 | $\checkmark$ |  |  |  |
| D3-08TD1 | $\checkmark$ |  |  |  |
| D3-08TD2 |  | $\checkmark$ |  |  |
| D3-16TD1-1 | $\checkmark$ | D3-16TD1-1 Discontinued 05/2022 |  |  |
| D3-16TD1-2* | $\checkmark$ |  |  |  |
| D3-16TD2 D3-1 | 6TD2 Discontinue | 06/2022 |  |  |
| D3-08TA-1 |  |  |  | $\checkmark$ |
| D3-08TA-2 | D3-08TA-2 Discontinued 09/2021 |  |  | $\checkmark$ |
| F3-16TA-1* |  |  |  | $\checkmark$ |
| F3-16TA-2 |  |  |  | $\checkmark$ |
| D3-16TA-2 |  |  |  | $\checkmark$ |
| D3-08TR |  |  | $\checkmark$ |  |
| F3-08TRS-1 |  |  | $\checkmark$ |  |
| F3-08TRS-2 |  |  | $\checkmark$ |  |
| F3-08TRS-4 |  |  | $\checkmark$ |  |
| D3-16TR |  |  | $\checkmark$ |  |
| D3-04TAS |  |  |  | $\checkmark$ |
| F3-08TAS* |  |  |  | $\checkmark$ |
| F3-08TAS-1 | F3-08TAS-1 Disco | ntinued 11/06/2023 |  | $\checkmark$ |

* Obsolete


## Configuration \#1 DL305 DC Current Sourcing Input Module



## Configuration \#2 DL305 DC Current Sinking/Sourcing Input Module



## Configuration \#3 DL305 DC Current Sinking Input Module



## Configuration \#4 DL305 AC/DC Input Module



## Configuration \#5 <br> DL305 AC Input Module



## Configuration \#6 DL305 DC Current Sinking Output Module

| Device Common | Load | Device Input | Sinking Output Module <br> Module <br> Output |
| :---: | :---: | :---: | :---: |
|  | Sourcing <br> Field Device |  |  |
|  |  | DeviceInput $\quad$ Current Flow $\longrightarrow$ |  |
| Device Common | Sourcing Field Device |  | Module Output |
|  |  | Current Flow $\longrightarrow$ |  |
|  | Load |  |  |
|  |  |  | Module Common |
|  |  |  |  |

## Configuration \#7 DL305 DC Current Sourcing Output Module



# Configuration \#8 DL305 AC/DC Current Sink/Source (Relay) Output Module 



## Configuration \#9 DL305 AC Output Module



## Solid State Field Device Wiring to DC Input Modules

NPN Field Device
Example


Current Sourcing Configuration

## PNP Field Device

## Example


(PNP) Current Sourcing Field Device

## Derating Characteristics

The DL305 input and output module operating specifications change depending on ambient temperature. The I/O specifications have a derating chart for each module which shows functionality in respect to ambient temperature.
The example below shows a derating curve for a D3-08ND2 discrete input module where the operating specifications do not change within the specified temperature operating range.


The example below shows a derating curve for a D3-16TD-1 discrete output module where the operating specifications are affected depending on ambient temperature.


## I/O Wiring Guidelines

General Considerations

The following information is to give you a general idea on how to wire the different types of modules in the DL305 system. For specific information on wiring a particular module refer to the specification sheet in the appropriate I/O chapter.
Consider the following guidelines when connecting the field wring.

1. There is a maximum AWG the modules can accept. You can uses a smaller AWG than is noted in the table below.

| Module type | Maximum AWG |
| :--- | :--- |
| 8 point | 12 |
| 16 point | 16 |

2. Always use a continuous length of wire, do not combine wires to attain a desired length.
3. Use the shortest possible cable length.
4. Use wire trays for routing where possible
5. Avoid running wires near high energy wiring.
6. Avoid running input wiring in close proximity to output wiring where possible.
7. To minimize voltage drops when wires must run a long distance , consider using multiple wires for the return line.
8. Avoid running DC wiring in close proximity to AC wiring where possible.
9. Avoid creating sharp bends in the wires.

Wiring the Different There are three main types of module faces for the DL305 I/O. These module faces Module Types are: lift covers over terminal blocks, flip covers over terminal blocks and D-shell compatible sockets. If the module you are using has a cover you can remove the cover either by lifting from the bottom or by flipping the door open. Some of the modules have removable terminal blocks. These modules can be recognized by the squeeze tabs on the top and bottom of the terminal block. To remove the terminal block, press the squeeze tabs and pull the block away from the module.

WARNING: For some modules, field device power may still be present on the terminal block even though the PLC system is turned off. To minimize the risk of electrical shock, check all field device power before you remove the connector.


## Fuse Protection

## External Fuse Example

To help avoid blowing the internal module fuses, we suggest you add external fuses to your I/O wiring. A fast blow fuse with a lower current rating than the I/O module fuse can be added to each common. Or, you can add a fuse with a rating of slightly less than the maximum current per output point to each output. Refer to the I/O module specification sheets to find the maximum current per point or per common for output modules. Adding the external fuse does not guarantee the prevention of module damage, but it will provide added protection.

> See Errata Sheet at the beginning of this file. The wording in this WARNING has been changed to: WARNING: Modules which have soldered-in or non-replaceable fuses are non-repairable and should be replaced with new modules.


WARNING: For modules which have soldered-in or non-replaceable fuses, we recommend that you return the module to us and let us replace your blown fuse(s) since the module fuses are attached to the board and disassembling the module will void your warranty.

## Discrete Input Modules

In This Chapter. . . .

- Discrete Input Module Identification and Terminology
- D3-08ND2, 24 VDC Input Module
- D3-16ND2-1, 24 VDC Input Module
- D3-16ND2-2, 24 VDC Input Module Module
- D3-16ND2F, 24 VDC Fast Response Input Module
- F3-16ND3F, TTL/24 VDC Fast Response Input Module
- D3-08NA-1, 110 VAC Input Module
- D3-08NA-2, 220 VAC Input Module
- D3-16NA, 110 VAC Input Module
— D3-08NE3, 24 VAC/DC Input Module
- D3-16NE3, 24 VAC/DC Input Module
- D3-08SIM, Input Simulator

D3-08NA-2 and D3-08ND2 Discontinued 06/23
D3-16NA Discontinued 10/2021

## Discrete Input Module Identification and Terminology

This chapter contains I/O specification sheets for the discrete input modules. The diagram below shows the status indicator location for some of the most common discrete input modules.

Discrete Input Module Status Indicators


Color Coding of I/O The DL305 family of I/O modules has a color coding scheme to help you identify Modules

Input Module Selection whether the module is an input module, an output module or a special module. This is done through a color bar indicator located on the front of each module below the part number. The following color scheme is used.


Module Type
Discrete/Analog Output Discrete/Analog Input Other

Color Code
Red
Blue
White

Your input module selection depends on the field devices used and system performance requirements. The input module specifications in this chapter list the information needed for choosing the correct module for a field device and to assure it meets the system requirements. The following list defines the specifications listed in this chapter.

Inputs Per Module Indicates number of input points per module and designates current sinking, current sourcing, or either.
Commons Per $\quad$ Number of commons per module and their electrical characteristics
Module

Input Voltage The operating voltage range of the input circuit. DL305 input modules require either Range an internal or external power supply for the operating voltage. The base power supply will provide the internal voltage.

Peak Voltage Maximum voltage allowed for the input circuit.
AC Frequency $\quad$ AC modules are designed to operate within a specific frequency range. 60 Hz is the standard AC frequency in the U.S., 50 Hz is common in other countries.

ON Voltage Level The voltage level at which the input point will turn ON.
OFF Voltage Level The voltage level at which the input point will turn OFF.
Input Current Typical operating current for an active (ON) input.
Input Impedance Input impedance can be used to calculate input current for a particular operating voltage.

## Minimum ON The minimum current for the input circuit to operate reliably in the ON state. Current <br> Maximum OFF

 CurrentBase Power
Required

OFF to ON
Response
ON to OFF
Response
Terminal Type
Power from the base power supply is used by the DL305 input modules and varies between different modules. The guidelines for using module power is explained in the power budget configuration section in chapter 4.

The time the module requires to process an OFF to ON state transition.

The time the module requires to process an ON to OFF state transition.

Indicates whether the terminal type is a removable or non-removable connector or terminal.

Status Indicators LEDs indicate the ON/OFF status of an input point. These LEDs are electrically located on either the logic side or the field device side of the input circuit.

Weight
Indicates the weight of the module. (See Appendix D for a complete listing of DL305 component weights.)

## D3-08ND2, 24 VDC Input Module D3-08TD2 Discontinued 06/2023

| Inputs per module | 8 (current sourcing) | Base power required | 9V 10 mA Max $24 \mathrm{~V} 14 \mathrm{~mA} / \mathrm{ON}$ pt. ( 112 mA Max) |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (internally connected) |  |  |
| Input voltage range | 18-36VDC |  |  |
| Input voltage | Internally supplied | OFF to ON response | $4-15 \mathrm{~ms}$ |
| Peak voltage | 40 VDC | ON to OFF response | $4-15 \mathrm{~ms}$ |
| AC frequency | N/A | Terminal type | Non-removable |
| ON voltage level | < 3 V | Status indicators | Field side |
| OFF voltage level | $>18 \mathrm{~V}$ | Weight | 4.2 oz. (120 g) |
| Input impedance | 1.8 K ohm |  |  |
| Input current | 12 mA Max |  |  |
| Minimum ON current | 7 mA |  |  |
| Maximum OFF current | 3 mA |  |  |



Derating Chart for D3-08ND2
Points



## D3-16ND2-1, 24 VDC Input Module <br> D3-16ND2-1 Discontinued 07/2021

$\left.\begin{array}{|l|l|l|l|l|}\hline \text { Inputs per module } & 16 \text { (current sourcing) } & & \text { Base power required } & \begin{array}{l}9 \mathrm{~V} 25 \mathrm{~mA} \mathrm{Max} \\ 24 \mathrm{~V} 14 \mathrm{~mA} / \mathrm{ON} \mathrm{pt.}\end{array} \\ \hline \text { Commons per module } & 224 \mathrm{~mA} \mathrm{Max)}\end{array}\right)$


## D3-16ND2-2, 24 VDC Input Module Module

| Inputs per module | 16 (current sourcing) | Base power required | $\begin{aligned} & 9 \mathrm{~V} 3 \mathrm{~mA}+1.3 \mathrm{~mA} / \mathrm{ON} \mathrm{pt} \\ & (24 \mathrm{~mA} \text { Max) } \\ & 24 \mathrm{~V} 1 \mathrm{~mA}+13 \mathrm{~mA} / \mathrm{ON} \mathrm{pt} \\ & (209 \mathrm{~mA} \text { Max) } \end{aligned}$ |
| :---: | :---: | :---: | :---: |
| Commons per module | 8 internally connected |  |  |
| Input voltage range | 18-36 VDC |  |  |
| Input voltage | Internally supplied |  |  |
| Peak voltage | 36 VDC | OFF to ON response | $4-15 \mathrm{~ms}$ |
| AC frequency | N/A | ON to OFF response | $4-15 \mathrm{~ms}$ |
| ON voltage level | < 3 V | Terminal type | 24 Pin Removable |
| OFF voltage level | > 19 V |  | connector |
| Input impedance | 2.2 K ohm | Status indicators | Field side |
| Input current | 20 mA Max | Weight | 5.3 oz. (150 g) |
| Minimum ON current | 5 mA |  |  |
| Maximum OFF current | 2 mA |  |  |



## D3-16ND2F, 24 VDC Fast Response Input Module

| Inputs per module | 16 (current sourcing) | Base power required | 9V 25 mA Max $24 \mathrm{~V} 14 \mathrm{~mA} / \mathrm{ON}$ pt. ( 224 mA Max) |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (internally connected) |  |  |
| Input voltage range | 18-36VDC |  |  |
| Input voltage | Internally supplied | OFF to ON response | 0.8 ms |
| Peak voltage | 36VDC | ON to OFF response | 0.8 ms |
| AC frequency | N/A | Terminal type | Removable |
| ON voltage level | < 13V | Status indicators | Field side |
| OFF voltage level | $>19 \mathrm{~V}$ | Weight | 6.3 oz. (180 g) |
| Input impedance | 1.8 K ohm |  |  |
| Input current | 20 mA Max |  |  |
| Minimum ON current | 5 mA |  |  |
| Maximum OFF current | 1 mA |  |  |



# F3-16ND3F, TTL/24 VDC Fast Response Input Module 

| Inputs per module | 16 sink/source (jumper selectable sink/source)* | Base power required | 9V 148 mA Max 24 V 68 mA Max |
| :---: | :---: | :---: | :---: |
|  |  | Input current | $\begin{aligned} & 1 \mathrm{~mA} @ \text { 5VDC } \\ & 3 \mathrm{~mA} @ 12-24 \mathrm{DC} \end{aligned}$ |
| Commons per module | 2 (non-isolated) | Input impedance | 4.7K |
| Input voltage range | $\begin{aligned} & 5 \text { VDC TTL \& CMOS, } \\ & 12-24 \text { VDC } \\ & \text { (jumper selectable)* } \end{aligned}$ | OFF to ON response | 1 ms |
|  |  | ON to OFF response | 1 ms |
| Input voltage supplied | Internal (used with sinking loads) External (used with sourcing loads) | Maximum input rate | 500 Hz |
|  |  | Minimum ON current | $\begin{aligned} & 0.4 \mathrm{~mA} @ 5 \mathrm{VDC} \\ & 0.9 \mathrm{~mA} @ 12-24 \mathrm{VDC} \end{aligned}$ |
| Peak voltage | 100 VDC (35 VDC Continuous) | Maximum OFF current | $\begin{aligned} & 0.8 \mathrm{~mA} @ 5 \mathrm{VDC} \\ & 2.2 \mathrm{~mA} @ 12-24 \mathrm{VDC} \end{aligned}$ |
| AC frequency | N/A | Terminal type | Removable |
| ON voltage level | $\begin{aligned} & 0-1.5 V D C @ 5 V D C \\ & 0-4 V D C @ 12-24 V D C \end{aligned}$ | Status indicators | Logic side |
| OFF voltage level | $\begin{aligned} & \text { 3.5-5VDC @ 5VDC } \\ & \text { 10-24VDC @12-24VDC } \end{aligned}$ | Weight | 5.4 oz. (153 g) |

* 12 Inputs are jumper selectable for 5VDC/12-24VDC and Sink Load/Source Load
4 Inputs are jumper selectable for 5VDC/12-24VDC and Sink Load/Source Load


Sinking Load Configuration


Derating Chart for F3-16ND3F



Jumper selected for sourcing load configuration. An external power supply must be used in this configuration.
The DC power to sense the state of the inputs when jumpers are installed for sinking type signals is provided by the rack power supply. Sinking type inputs are turned ON by switching the input circuit to common. Source type input signals assume the ON state until the input device provides the voltage to turn the input OFF.

## Selection of Operating Mode:

The mode of operation, either 5VDC or 12-24VDC sink or source, for each group of circuits is determined by the position of jumper plugs on pins located on the edge of the circuit board. There are four sets of pins (3 pins in each set), with two sets for each group of inputs. The first two sets of pins are used to configure the first 12 inputs (eg. 0 to 7 and 100 to 103) and are labeled 12 CIRCUITS. Above the first set of pins are the labels $12 / 24 \mathrm{~V}$ and 5 V . Above the second set of pins are the labels SINK and SRC (source). To select an operating mode for the first 12 circuits, place a jumper on the two pins nearest the appropriate labels. For example, to select 24VDC Sink input operation for the first 12 inputs, place a jumper on the two pins labeled $12 / 24 \mathrm{~V}$ and on the two pins labeled SINK. The last two sets of pins are used to configure the last 4 inputs (eg. 104 to 107) and are labeled 4 CIRCUITS. The operating mode selected for the last group of 4 inputs can be different than the mode chosen for the first group of 12 inputs. Correct module operation requires each set of three pins have a jumper installed (four jumpers total).

NOTE: When a group of inputs are used with TTL logic, select the SINK operating mode for that group. "Standard" TTL can sink several milliamps but can source less than 1 mA .

# D3-08NA-1, 110 VAC Input Module 

D3-08NA-1 Discontinued 04/08/2022

| Inputs per module | 8 | Minimum ON current | 8 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Maximum OFF current | 2 mA |
| Input voltage range | 85-132VAC | Base power required | 9V 10 mA Max |
| Input voltage supply | External |  |  |
| Peak voltage | 132VAC | OFF to ON response | $10-30 \mathrm{~ms}$ |
| AC frequency | $47-63 \mathrm{~Hz}$ | ON to OFF response | $10-60 \mathrm{~ms}$ |
| ON voltage level | >80 VAC | Terminal type | Non-removable |
| OFF voltage level | <20 VAC | Status indicators | Field side |
| Input impedance | 10 K ohm | Weight | 5 oz ( 140 g ) |
| Input current | $\begin{aligned} & 15 \mathrm{~mA} @ 50 \mathrm{~Hz} \\ & 18 \mathrm{~mA} @ 60 \mathrm{~Hz} \end{aligned}$ |  |  |
|  |  |  |  |
|  |  |  |  |



## D3-08NA-2, 220 VAC Input Module

| Inputs per module | 8 |  | Minimum ON current | 10 mA |
| :--- | :--- | :--- | :--- | :--- |
|  | Commons per module | 2 (isolated) |  | Maximum OFF current | 2 mA



## D3-16NA, 110 VAC Input Module D3-16NA Discontinued 10/2021

| Inputs per module | 16 | Minimum ON current | 8 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Maximum OFF current | 1.5 mA |
| Input voltage range | 80-132VAC | Base power required* | 9V 6.25 mA Max/ON pt. |
| Input voltage supply | External |  |  |
| Peak voltage | 132VAC | OFF to ON response | $5-50 \mathrm{~ms}$ |
| AC frequency | $50-60 \mathrm{~Hz}$ | ON to OFF response | $5-60 \mathrm{~ms}$ |
| ON voltage level | >80 VAC | Terminal type | Removable |
| OFF voltage level | <15 VAC | Status indicators | Logic side |
| Input impedance | 8 K ohm | Weight | 6.4 oz. (180 g) |
| Input current | $\begin{aligned} & 16 \mathrm{~mA} @ 50 \mathrm{~Hz} \\ & 25 \mathrm{~mA} @ 60 \mathrm{~Hz} \end{aligned}$ |  |  |
|  |  |  |  |
|  |  |  |  |

* 9 V typical values are $4 \mathrm{~mA} / \mathrm{ON}$ pt., 64 mA total



## D3-08NE3, 24 VAC/DC Input Module

| Inputs per module | 8 (sink/source) | Base power required | $\begin{aligned} & 9 \mathrm{~V} 10 \mathrm{~mA} \max \\ & 24 \mathrm{~V} / \mathrm{N} / \mathrm{A} \end{aligned}$ |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) |  |  |
| Input voltage range | 20-28 VAC/VDC | OFF to ON response | AC: $5-50 \mathrm{~ms}$ DC: $6-30 \mathrm{~ms}$ |
| Input voltage | External |  |  |
| Peak voltage | 28 VAC/VDC | ON to OFF response | AC/DC: 5-60 ms |
| AC frequency | $47-63 \mathrm{~Hz}$ | Terminal type | Non-removable |
| ON voltage level | >20 V | Status indicators | Field side |
| OFF voltage level | <6V | Weight | 4.2 oz. (120 g) |
| Input impedance | 1.5 K ohm |  |  |
| Input current | 19 mA Max |  |  |
| Minimum ON current | 10 mA |  |  |
| Maximum OFF current | 2 mA |  |  |



## D3-16NE3, 24 VAC/DC Input Module

| Inputs per module | 16 (sink/source) |  | Base power required | $9 \mathrm{~V} 2.5 \mathrm{mA}+.4.5 \mathrm{~mA} /$ <br> ON pt.(130 mA max) <br> $24 \mathrm{~V} \mathrm{~N} / \mathrm{A}$ |
| :--- | :--- | :--- | :--- | :--- |
| Commons per module | 2 (isolated) |  |  |  |



## D3-08SIM, Input Simulator

| Inputs per module | 8 |  |  |  |
| :--- | :--- | :--- | :--- | :--- |
| Base Power required | $10 \mathrm{~mA} \mathrm{@} \mathrm{9VDC}$ <br> 112 mA max @ <br> 24 VDC |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| OFF to ON response | $4-15 \mathrm{~ms}$ |  |  |  |
| ON to OFF response | $4-15 \mathrm{~ms}$ |  |  |  |
| Terminal type | None |  |  |  |
| Status indicators | Switch side |  |  |  |
| Weight | 3.0 oz. $(85 \mathrm{~g})$ |  |  |  |



## Discrete Output Modules

In This Chapter. . . .
— Discrete Output Module Identification and Terminology

- Relay Arc Suppression - DC and AC Applications
- D3-08TD1, 24 VDC Output Module
- D3-08TD2, 24 VDC Output Module
- D3-16TD1-1, 24 VDC Output Module
- D3-16TD1-2, 24 VDC Output Module
- D3-16TD2, 24 VDC Output Module
- D3-04TAS, 110-220 VAC Output Module
- F3-08TAS, 250 VAC Isolated Output Module
- F3-08TAS-1, 125 VAC Isolated Output Module
— D3-08TA-1, 110-220 VAC Output Module
- D3-08TA-2, 110-220 VAC Output Module
- F3-16TA-2, 20-125 VAC Output Module
- D3-16TA-2, 110-220 VAC Output Module
- D3-08TR, Relay Output Module
- F3-08TRS-1, Relay Output Module
- F3-08TRS-2, Relay Output Module
- D3-16TR, Relay Output Module

D3-16TD2Discontinued06/2022.
D3-16TD1-1Discontinued05/2022PleaseconsiderBRX,ProductivityorCLICKsystemsas areplacementPLCplatform.
D3-08TA-1Discontinued01/2022PleaseconsiderBRX,ProductivityorCLICKsystemsasa replacementPLCplatform.
D3-08TRmodulehasbeenretiredasof03/2022.PleaseconsiderBRX,Productivityor
CLICKsystemsasareplacementPLCplatform.

# Discrete Output Module Identification and Terminology 

Discrete Output Module Status Indicators

This chapter contains I/O specification sheets for the DL305/FL305 discrete output modules. The following diagram shows the status indicator location for some of the most common discrete output modules.


Color Coding of I/O The DL305 family of I/O modules has a color coding scheme to help you identify Modules

Output Modules Selection whether the module is an input module, an output module or a special module. This is done through a color bar indicator located on the front of each module below the part number. The following color scheme is used.


Color Code
Red
Blue
White

Your output module selection depends on the field devices used and system performance requirements. The output module specifications in this chapter list the information which is needed for choosing the correct module for a field device and to assure it meets the system requirements. The following list defines the specifications listed in this chapter.

Outputs Per
Module

Module
Operating Voltage
Output Type
(Resistive)
Maximum Leakage
Current
Maximum Inrush
Current

Minimum Load
Base Power Required

OFF to ON
Response Time
ON to OFF
Response Time
Terminal Type

Commons Per Number of commons per module and their electrical characteristics.

Peak Voltage Maximum voltage the output circuit can control.
AC Frequency $\quad$ AC modules are designed to operating within a specific frequency range. 60 Hz is the standard AC frequency in the U.S., 50 Hz is common in other countries.

ON Voltage Drop The voltage between the output point and common during an active ON with a load.
Maximum Current The maximum current for an output with a resistive load.
Indicates number of output points per module and designates current sinking, current sourcing, or either.

The operating voltage range of the output circuit.
The output circuit can be a transistor, a triac, or a relay. The NPN or PNP transistor outputs are normally used in low voltage or high speed DC applications. Triac outputs are used in AC voltage applications. The Form A or C relay outputs are normally used where a wide voltage range is needed. Relay output modules are capable of carrying more current than a transistor or a triac output and can pass AC or DC voltages. The disadvantage of a relay module is the internal power consumption and the relay life expectancy.

The maximum current of the output circuit during an OFF state.
The maximum current over a short period of time during the OFF to ON transition of a output point. It is greater than the normal ON state current and depends on the field device electrical characteristics.

The minimum load across the output's circuit for the circuit to operate properly.
Power from the base power supply is used by the DL305 output modules and varies between different modules. The guidelines for using module power is explained in the power budget configuration section in chapter 4.

The processing time the module requires to transition from an OFF to ON state.
The processing time the module requires to transition from an ON to OFF state.
Indicates whether the terminal type is a removable or non-removable connector or terminal.

LEDs indicate the ON/OFF status of an input point. These LEDs are electrically located on either the logic side or the field device side of the output circuit.

Indicates the current rating of the replaceable or non-replaceable fuse(s).
Amount of closures typical for a relay point before failure.
Indicates the weight of the module.

## Relay Arc Suppression - DC and AC Applications

FL305 High Current This application note describes the addition of external contact protection to high

Relay Output
Module Arc
Suppression

Resistor and
Capacitor Selection current isolated relay output modules. It supplements the wiring information for the F3-08TRS-1 and F3-08TRS-2 relay output modules.
Adding external contact protection may extend a relays life beyond the number of operations listed. High current inductive loads such as clutches, brakes, motors, direct acting solenoid valves, and motor starters will benefit the most from external contact protection.

$$
\begin{aligned}
& \mathrm{C}(\mu \mathrm{~F})=\mathrm{I}^{2} / 10 \\
& \mathrm{R}(\Omega)=\mathrm{V} / 10 \mathrm{I}^{\mathrm{x}} \quad \text { where } \mathrm{x}=(1+50 / \mathrm{E})
\end{aligned}
$$

Use peak AC values for I and V, see "Peak Voltage and Current" below. Where I = Amperes of load current immediately prior to opening of contacts. Where $\mathrm{E}=$ Source voltage immediately prior to closing of contacts.

$$
\begin{aligned}
& \text { R minimum }=0.5 \Omega, 1 / 2 \mathrm{~W} \\
& \mathrm{C} \text { minimum }=0.001 \mu \mathrm{~F} \text {, the voltage rating of } \mathrm{C} \text { must be } \geq \mathrm{E}
\end{aligned}
$$

Resistor Tolerance For E $<70 \mathrm{~V}, \mathrm{R}$ may be 3 times indicated value.
For $70 \mathrm{~V}<\mathrm{E}<100 \mathrm{~V}$, $R$ may be $\pm 50 \%$ indicated value.
For $100 \mathrm{~V}<\mathrm{E}<150 \mathrm{~V}, \mathrm{R}$ may be $\pm 10 \%$ indicated value.
For $E>150 V$, $R$ may be $\pm 5 \%$ indicated value.
Peak Voltage and The following equations can be used to determine $I_{\text {peak }}$ and $V_{\text {peak }}$ :

## Current

$$
\begin{aligned}
\mathrm{I}_{\text {peak }} & =\mathrm{I}_{\mathrm{rms}} / .707 \\
\mathrm{~V}_{\text {peak }} & =\mathrm{V}_{\mathrm{rms}} / .707 \\
\mathrm{I}_{\text {peak }} & =\mathrm{I}_{\text {ave }} / .636
\end{aligned}
$$

Alternating Current

DC Rectified Alternating Current
If the contact is switching a DC inductive load, add a diode across the load as near to load coil as possible. Add the RC network across the relay contacts as close to the relay as possible.

## Resistor and Capacitor Nomogram

The nomogram shown below affords a convenient method of selecting the proper contact protection for P \& B relays used in F3-08TRS-1 and F3-08TRS-2 modules.
Example: Use a current (1) of 1.0 ampere and a voltage ( E ) of 50 volts.
Capacitance ( $C$ ) in microfarads is found directly on the left side scale, opposite 1.0 amperes as 0.1 . Resistance (R) in ohms is obtained using a straight edge. Locate 1.0 amperes (I) on the left side scale and 50 volts (E) on the center scale. Place the straightedge on these points. The junction of the straight edge and the right side determines R. In this example $R$ is equal to 5.0 ohms.


For DC. For AC, use peak values
(2) $R=E / 10 I^{\times}$ohms

Where $x=(1+50 / E)$

## D3-08TD1, 24 VDC Output Module

| Outputs per module | 8 (current sinking) | Minimum load | 1 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (internally connected) | Base power required | 9 V 20 mA Max |
| Operating voltage | 5-24VDC |  | $24 \mathrm{~V} 3 \mathrm{~mA} / \mathrm{pt}$. |
| Output type | NPN (open collector) |  | (24mA Max) |
|  |  | OFF to ON response | 0.1 ms |
| Peak voltage | 45VDC | ON to OFF response | 0.1 ms |
| AC frequency | N/A | Terminal type | Non-removable |
| ON voltage drop | 0.8V @ 0.5A | Status indicators | Logic Side |
| Max current | 0.5A / point 1.8 / common | Weight | 4.2 oz. (120 g) |
|  |  | Fuses |  |
| Max leakage current | 0.1 mA @ 40VDC |  | One 3A per common |
| Max inrush current | $\begin{aligned} & 3 \mathrm{~A} / 20 \mathrm{~ms} \\ & 1 \mathrm{~A} / 100 \mathrm{~ms} \end{aligned}$ |  | Non-replaceable |



## D3-08TD2, 24 VDC Output Module

| Outputs per module | 8 (current sourcing) | Minimum load | 1 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (internally connected) | Base power required | $\begin{aligned} & 9 \mathrm{~V} 30 \mathrm{~mA} \text { Max } \\ & 24 \mathrm{~V} \text { N/A } \end{aligned}$ |
| Operating voltage | 5-24VDC |  |  |
| Output type | NPN Transistor (emitter follower) | OFF to ON response | 0.1 ms |
|  |  | ON to OFF response | 0.1 ms |
| Peak voltage | 40VDC | Terminal type | Non-removable |
| AC frequency | N/A | Status indicators | Logic Side |
| ON voltage drop | 1V @ 0.5A | Weight | 4.2 oz. (120 g) |
| Max current | 0.5A / point 1.8A/ common | Fuses | (2) <br> One 3A per common |
| Max leakage current | 0.1 mA @ 24VDC |  | Non-replaceable |
| Max inrush current | $\begin{aligned} & 3 \mathrm{~A} / 20 \mathrm{~ms} \\ & 1 \mathrm{~A} / 100 \mathrm{~ms} \end{aligned}$ |  |  |



## D3-16TD1-1, 24 VDC Output Module D3-16TD1-1 Discontinued 05/2022

| Outputs per module | 16 (current sinking) | Minimum load | 1 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (internally connected) | Base power required | 9V (40 mA Max) $3 \mathrm{~mA}+2.3 \mathrm{~mA} / \mathrm{ON}$ pt. $24 \mathrm{~V} 6 \mathrm{~mA} / \mathrm{ON}$ pt. ( 96 mA Max) |
| Operating voltage | 5-24VDC |  |  |
| Output type | NPN transistor (open collector) |  |  |
| Peak voltage | 45VDC | OFF to ON response | 0.1 ms |
| AC frequency | N/A | ON to OFF response | 0.1 ms |
| ON voltage drop | 2V @ 0.5A | Terminal type | Removable |
| Max current | 0.5A/point 2A/ common | Status indicators | Logic Side |
|  |  | Weight | 5.6 oz. (160 g) |
| Max leakage current | 0.1 mA @ 40VDC | Fuses | (2) |
| Max inrush current | $\begin{aligned} & 3 \mathrm{~A} / 20 \mathrm{~ms} \\ & 1 \mathrm{~A} / 100 \mathrm{~ms} \end{aligned}$ |  | One 3A per common Non-replaceable |



## D3-16TD1-2, 24 VDC Output Module

| Outputs per module | 16 (current sinking) | Minimum load | 1 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 4 (internally connected) | Base power required | 9 V (40mA Max) |
| Operating voltage | 5-24VDC |  | $3 \mathrm{~mA}+2.3 \mathrm{~mA} / \mathrm{ON}$ pt. 24 V 6mA/ON pt. |
| Output type | NPN transistor (open collector) |  | (96mA Max) |
|  |  | OFF to ON response | 0.1 ms |
| Peak voltage | 45VDC | ON to OFF response | 0.1 ms |
| AC frequency | N/A | Terminal type | Removable connector |
| ON voltage drop | 2.0V @ 0.5A | Status indicators | Logic Side |
| Max current | 0.5A / point 1.8A common | Weight | 5.6 oz. (160 g) |
|  |  | Fuses |  |
| Max leakage current | 0.3 mA @ 40VDC |  | One 3A per common |
| Max inrush current | 3A / 20ms <br> 1A / 100ms |  |  |



## D3-16TD2, 24 VDC Output Module

| Outputs per module | 16 (current sourcing) | Minimum load | 1 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Base power required | 9V $7.5 \mathrm{~mA} / \mathrm{ON}$ pt. ( 180 mA Max) 24V N/A |
| Operating voltage | 5-24VDC |  |  |
| Output type | NPN transistor (emitter follower) |  |  |
|  |  | OFF to ON response | 0.1 ms |
| Peak voltage | 40VDC | ON to OFF response | 1 ms |
| AC frequency | N/A | Terminal type | Removable |
| ON voltage drop | 1.5V @ 0.5A | Status indicators | Logic Side |
| Max current | 0.5A / point 3A common | Weight | 7.1 oz. (210 g) |
|  |  | Fuses |  |
| Max leakage current | $0.01 \mathrm{~mA} @ 40 \mathrm{VDC}$ |  | One 5A per common |
| Max inrush current | $\begin{aligned} & 3 \mathrm{~A} / 20 \mathrm{~ms} \\ & 1 \mathrm{~A} / 100 \mathrm{~ms} \end{aligned}$ |  | Non-replaceable |



## D3-04TAS, 110-220 VAC Output Module

| Outputs per module | 4 | Minimum load | 10 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 4 (isolated) | Base power required | 9V 12 mA Max 24 V N/A |
| Operating voltage | 80-265VAC |  |  |
| Output type | Triac | OFF to ON response | 1 ms Max |
| Peak voltage | 265 VAC | ON to OFF response | 10 ms Max |
| AC frequency | $47-63 \mathrm{~Hz}$ | Terminal type | Non-removable |
| ON voltage drop | 1.5 VAC @ 2A | Status indicators | Logic Side |
| Max current | 2A / point <br> 2A/ common | Weight | 6.4 oz. (180 g) |
|  |  | Fuses |  |
| Max leakage current | $\begin{aligned} & 7 \mathrm{~mA} @ 220 \mathrm{VAC} \\ & 3.5 \mathrm{~mA} @ 110 \mathrm{VAC} \end{aligned}$ |  | One 3A per common User replaceable |
| Max inrush current | 20A for 16 ms 10 A for 100 ms |  |  |



## F3-08TAS, 250 VAC Isolated Output Module

| Outputs per module | 8 (500V point-to-point isolation) | Base power required | 9V $10 \mathrm{~mA} / \mathrm{ON}$ pt. 80mA Max. 24V N/A |
| :---: | :---: | :---: | :---: |
| Commons per module | 8 (isolated) |  |  |
| Operating voltage | 12-125 VAC 125-250 VAC requires external fuses | OFF to ON response | 8 ms Max |
|  |  | ON to OFF response | 8 ms Max |
| Output type | SSR Array (TRIAC) | Terminal type | Removable |
| Peak voltage | 400 VAC | Status indicators | Logic Side |
| AC frequency | $47-440 \mathrm{~Hz}$ | Weight | 6.3 oz. (177g) |
| ON voltage drop | 1 VAC @ 1A | Fuses | (8) fast blow |
| Max current | 1A / point | BK/PCE-5 Bussman | One 5A (125V fast |
| Max leakage current | $10 \mu \mathrm{~A} @ 240$ VAC | (One spare fuse included) | blow) per each circuit User replaceable |
| Max inrush current* | 20A for 16 ms $3 A$ for 100 ms |  |  |
| Minimum load | 0.5 mA |  |  |



## F3-08TAS-1, 125 VAC Isolated Output Module

| Outputs per module | 8 (1500V point-topoint isolation) | Base power required | $9 \mathrm{~V} 25 \mathrm{~mA} / \mathrm{ON}$ pt. (200mA Max), 24V N/A |
| :---: | :---: | :---: | :---: |
| Commons per module | 8 (isolated) |  |  |
| Operating voltage | 20-125VAC | OFF to ON response | 1 ms Max |
|  |  | ON to OFF response | 9 ms Max |
| Output type | SSR (TRIAC with zero cross-over) | Terminal type | Removable |
| Peak voltage | 140VAC | Status indicators | Logic Side |
| AC frequency | $47-63 \mathrm{~Hz}$ | Weight | 6.3 oz. (177g) |
| ON voltage drop | 1.6 V (rms) @ 1.5A | Fuses | 8 (1 per common) 5A, 125V fast blow Order D3-FUSE-4 (5 per pack) |
| Max current | 1.5A/point |  |  |
| Max leakage current | 0.7 mA (rms) |  |  |
| Max inrush current* | 15A for 20 ms 8 A for 100 ms |  |  |
| Minimum load | 50 mA |  |  |



## D3-08TA-1, 110-220 VAC Output Module

| Outputs per module | 8 | Minimum load | 25 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Base power required | $9 \mathrm{~V} 20 \mathrm{~mA} / \mathrm{ON}$ pt. ( 160 mA Max) 24 V N/A |
| Operating voltage | 80-265VAC |  |  |
| Output type | Triac |  |  |
| Peak voltage | 265VAC | OFF to ON response | 1 ms Max |
| AC frequency | $47-63 \mathrm{~Hz}$ | ON to OFF response | 8.33 ms Max |
| ON voltage drop | 1.5 VAC @ 1A | Terminal type | Removable |
| Max current | 1A / point <br> 3A/ common | Status indicators | Logic Side |
|  |  | Weight | 7.4 oz. (210 g) |
| Max leakage current | $\begin{aligned} & 1.2 \mathrm{~mA} @ 220 \mathrm{VAC} \\ & 0.52 \mathrm{~mA} @ 110 \mathrm{VAC} \end{aligned}$ | Fuses | (2) One 5A per common |
| Max inrush current | 10A for 16 ms 5 A for 100 ms |  | Non-replaceable |



D3-08TA-2 is discontinued as of 09/2021. Please consider BRX,

## D3-08TA-2, 110-220 VAC Output Module

| Outputs per module | 8 |  | Base power required | $9 \mathrm{~V} 20 \mathrm{~mA} / \mathrm{ON} \mathrm{pt}$. <br> $(160 \mathrm{~mA} \mathrm{Max})$ <br> $24 \mathrm{~V} \mathrm{N/A}$ |
| :--- | :--- | :--- | :--- | :--- |
| Commons per module | 2 (isolated) |  |  |  |



Derating Chart for D3-08TA-2 Points



## F3-16TA-2, 20-125 VAC Output Module D3-16TA-2 Retired 06/2023

| Outputs per module | 16 | Minimum load | 50 mA |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Base power required | $9 \mathrm{~V} 14 \mathrm{~mA} / \mathrm{ON} \mathrm{pt}$. |
| Operating voltage | 20-125VAC |  | 250mA Max. $24 \mathrm{~V} \text { N/A }$ |
|  |  | OFF to ON response | 8ms Max. |
|  |  | ON to OFF response | $8 \mathrm{~ms} \mathrm{Max}$. |
| Output type | SSR Array (TRIAC) | Terminal type | Removable |
| Peak voltage | 140 VAC | Status indicators | Logic Side |
| AC frequency | $47-63 \mathrm{~Hz}$ | Weight | 7.7oz. (218g) |
| ON voltage drop | 1.1VAC @ 1.1A | Fuses | 4 (One 5A 125V fast |
| Max current | 1.1A / point | (One spare fuse | blow per each group |
| Max leakage current | 0.7mA @ 125VAC | inclu | Order D3-FUSE-4 |
| Max inrush current* | 15A for 20 ms 8 A for 100 ms |  | (5 per pack) |



## D3-16TA-2, 110-220 VAC Output Module

| Outputs per module | 16 | Minimum load | 10 mA @ 15VAC |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Base power required * | $\begin{aligned} & 9 \mathrm{~V} 25 \mathrm{~mA} \text { Max /ON pt. } \\ & 400 \mathrm{~mA} \text { Max } \\ & 24 \mathrm{~V} \text { N/A } \end{aligned}$ |
| Operating voltage | 15-265 VAC |  |  |
| Output type | Triac |  |  |
| Peak voltage | 265 VAC | OFF to ON response | 1 ms Max |
| AC frequency | $47-63 \mathrm{~Hz}$ | ON to OFF response | 9 ms Max |
| ON voltage drop | 1.5 VAC @ 0.5A | Terminal type | Removable |
| Max current | $0.5 \mathrm{~A} /$ point 3A/ common 6A / per module | Status indicators | Logic Side |
|  |  | Weight | 7.2 0z. (210 g) |
|  |  | Fuses | (2) <br> One 5A per common Non-replaceable |
| Max leakage current | 4 mA @ 265 VAC |  |  |
| Max inrush current | 10 A for 10 ms 5 A for 100 ms |  |  |

* 9 V typical values $17 \mathrm{~mA} / \mathrm{ON}$ pt., 272 mA total




## D3-08TR, Relay Output Module

| Outputs per module | 8 | Minimum load | 5 mA @ 5v |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Base power required | $9 \mathrm{~V} 45 \mathrm{~mA} / \mathrm{ON}$ pt. ( 360 mA Max) 24 V N/A |
| Operating voltage | $\begin{aligned} & 5-265 \mathrm{VAC} \\ & 5-30 \mathrm{VDC} \end{aligned}$ |  |  |
| Output type | Form A (SPST) | OFF to ON response | 5 ms |
| Peak voltage | 265VAC / 30VDC | ON to OFF response | 5 ms |
| AC frequency | $47-63 \mathrm{~Hz}$ | Terminal type | Non-removable |
| ON voltage drop | N/A | Status indicators | Logic Side |
| Max current | 4A / point AC | Weight | $7 \mathrm{oz} .(200 \mathrm{~g})$ |
|  | 5A / point DC 6A / common | Fuses | (2) One 10A per common User replaceable |
| Max leakage current | 1 mA @ 220VAC |  |  |
| Max inrush current | 5A |  |  |

Typical Relay Life (Operations) Voltage Resistive Solenoid Closures

| 220VAC | 4 A | 0.5 A | 100 k |
| :--- | :---: | :---: | :---: |
| 220VAC |  | 0.05 A | 800 k |
| 110VAC | 4 A | 0.5 A | 100 k |
| 110VAC |  | 0.1 A | 650 k |
| 24VDC | 5 A | 0.5 A | 100 k |




Derating Chart for D3-08TR



D3-08TR module has been retired as of 03/2022. Please consider BRX, Productivity or CLICK systems as a replacement PLC platform.

## F3-08TRS-1, Relay Output Module

| Outputs per module | 8 | Max leakage current | N/A |
| :---: | :---: | :---: | :---: |
| Commons per module | 8 (isolated) | Max inrush current | 10A Inductive |
| Operating voltage* | 12-125 VAC 125-250 VAC requires external fuses 12-30 VDC | Minimum load | 100 mA @12VDC |
|  |  | Base power required | 9V 37mA / ON pt. (296 mA Max) 24 V N/A |
| Output type | 6 Form A (SPST) 2 Form C (SPDT) | OFF to ON response | 13 ms Max |
|  |  | ON to OFF response | 9 ms Max |
| Peak voltage | 265 VAC / 120 VDC | Terminal type | Removable |
| AC frequency | $47-63 \mathrm{~Hz}$ | Status indicators | Logic Side |
| ON voltage drop | N/A | Weight | 8.9 oz. (252 g) |
| Max current (resistive) | 10A / point AC/DC 30A / module AC/DC | Fuses | (8) One 10A (125V) per common Non-replaceable |

NOTE: Contact life may be lengthened beyond those values shown by the use of an appropriate arc suppression. This technique is discussed earlier in this chapter.


## F3-08TRS-2, Relay Output Module

| Outputs per module | 8 |
| :--- | :--- |
| Commons per module | 8 (isolated) |
| Operating voltage* | $12-125 ~ V A C$ <br> $125-250 ~ V A C ~ r e q u i r e s ~$ <br> external fuses <br> $12-30 ~ V D C ~$ |
|  | 6 Form A (SPST) <br> 2 Form C (SPDT) |
| Output type | 265 VAC / 120 VDC |
| Peak voltage | $47-63 \mathrm{~Hz}$ |
| AC frequency | $\mathrm{N} / \mathrm{A}$ |
| ON voltage drop | $5 \mathrm{~A} /$ point AC/DC <br> $40 \mathrm{~A} /$ module AC/DC |
| Max current (resistive) |  |
|  |  |


| Max leakage current | N/A |
| :--- | :--- |
| Max inrush current | 10 A Inductive |
| Minimum load | 100 mA @12VDC |
| Base power required | $9 \mathrm{~V} 37 \mathrm{~mA} / \mathrm{ON}$ pt. <br> $(296 \mathrm{~mA}$ Max) <br> $24 \mathrm{~V} \mathrm{N/A}$ |
| OFF to ON response | 13 ms Max |
| ON to OFF response | 9 ms Max |
| Terminal type | Removable |
| Status indicators | Logic Side |
| Weight | 9 oz. (255 g) |
| Fuses <br> 19379-K-10A <br> Wickman | (8) One 5A (125V) per <br> common <br> User replaceable |

NOTE: Contact life may be lengthened beyond those values shown by the use of an appropriate arc suppression. This technique is discussed earlier in this chapter.


## D3-16TR, Relay Output Module

| Outputs per module | 16 | Minimum load | 5 mA @ 5v |
| :---: | :---: | :---: | :---: |
| Commons per module | 2 (isolated) | Base power required | $9 \mathrm{~V} 30 \mathrm{~mA} / \mathrm{ON}$ pt. ( 480 mA Max) 24 V N/A |
| Operating voltage | $\begin{aligned} & 5-265 \text { VAC } \\ & 5-30 \text { VDC } \end{aligned}$ |  |  |
| Output type | 16 Form A (SPST) | OFF to ON response | 12 ms |
| Peak voltage | 265 VAC / 30 VDC | ON to OFF response | 12 ms |
| AC frequency | $47-63 \mathrm{~Hz}$ | Terminal type | Removable |
| ON voltage drop | N/A | Status indicators | Logic Side |
| Max current | 2A / point AC/DC (resistive) | Weight | 8.5 oz. (248g) |
|  |  | Fuses | None |
| Max leakage current | 0.1 mA @ 220 VAC |  |  |
| Max inrush current | 2A |  |  |

Typical Relay Life (Operations)

| Voltage | Resistive Solenoid Closures |  |  |
| :--- | :--- | :--- | ---: |
| 220VAC | 2 A | 0.25 A | 100 k |
| 220VAC |  | 0.03 A | 800 k |
| 110VAC | 2 A | 0.25 A | 100 k |
| 110VAC |  | 0.05 A | 650 k |
| 24VDC | 2 A | 0.25 A | 100 k |



Derating Chart for D3-16TR



## System Operation

In This Chapter. . . .

- Introduction
- CPU Operating System
- Initial Mode Setting and Memory Initialization
- Program Mode Operation
- Run Mode Operation
- I/O Response Time
- CPU Scan Time Considerations
- Memory Map
— I/O Point Bit Map
- Control Relay Bit Map
- Special Relays
- Timer / Counter Registers and Contacts
- Data Registers
- Stage Control / Status Bit Map
- Shift Register Bit Map
- Special Registers


## Introduction

Achieving the proper control for your equipment or process requires a good understanding of how the DL305 CPUs control all aspects of the system operation. This includes many things, such as I/O updates, program execution, etc. Take a few minutes to understand how the CPU stores and processes information. For more complex applications, this knowledge will make it easier to program and debug your application program to meet your system performance requirements.
There are four primary things you need to understand before you create your application program

- CPU Operating System - the CPU is the heart of the system. It manages all aspects of system control. A quick overview of all the steps is provided in the next section.
- CPU Operating Modes - the CPU has different operating modes that allow different types of operations. There are two primary modes of operation, Program Mode and Run Mode.
- CPU Timing - it is important you understand how the CPU timing can affect the system operation. The two most important areas are the I/O response time and the CPU scan time.
- CPU Memory Map - The DL305 CPUs offer a wide variety of memory types, such as timers, counters, inputs, etc. It is important to understand what memory types are available and how the memory areas are organized.
The remainder of this chapter discusses these items in detail.


## CPU Operating System

After the initial power-up sequence, the DL305 CPUs process data cyclically. There are several tasks the CPU must perform during each cycle, such as updating the I/O status, servicing external communications, executing the application program, etc. There are many different segments of execution, but the overwhelming majority of your concerns will be with the portion of the execution cycle that occurs during Run Mode. These operations will be discussed throughout the remainder of this chapter.

## DL305 CPU Operational Flow Chart

System Operation

## Initial Mode Setting and Memory Initialization

Flow Chart for
The previous flowchart contained a step called Initial Mode Setting. Once power has Initial Mode Setting been connected to the system, the CPU executes the following procedure to (\#1) determine which mode of operation should be entered.


## Memory Initialization

Before the CPU can begin normal operation, all memory areas are initialized. The CPU completes the following operations to initialize the memory areas.

| Item | Procedure |
| :--- | :--- |
| I/O Points | Input Points: activated, CPU will monitor status. <br> Output Points: turned off |
| Control Relays | Non-Retentive: turned off <br> Retentive: retains the condition prior to the system <br> power interruption. |
| Shift Registers | Retains the condition prior to the system power <br> interruption. |
| Timer / Counter Current Values | Timers: reset to zero <br> Counters: retains the current count prior to the <br> system power interruption. |
| Stages | Non-Retentive: turned off <br> Retentive: retains the condition prior to the system <br> power interruption. |
| Data Registers | Retains the condition prior to the system power <br> interruption. |

NOTE: Not all memory areas are retentive by default. See Chapter 3 for details on how to set the CPU to have retentive memory. Also, the memory map section provided later in this chapter shows the exact ranges that can be selected as retentive.

## Program Mode Operation

In Program Mode, the CPU does not execute the application program or update the output modules. The primary use for Program Mode is to enter or change an application program. You can also use the program mode to set up CPU parameters, such as the network address for the communication port on the DL340, retentive memory areas, etc.


You can use the Handheld Programmer key switch to select Program Mode operation, or you can use a DirectSOFT menu option to change the CPU mode.

## Run Mode Operation

In Run Mode, the CPU executes the application program and updates the I/O system. You can perform many operations during Run Mode. Some of these include:

- Monitor and change I/O point status
- Update timer/counter preset values
- Update Register memory locations

Even though there are many steps in the overall flowchart of operation, the Run Mode operation can be divided into a few key areas. It is very important you understand how each of these areas of execution can affect the results of your application program solutions.


Update I/O Points The CPU reads the status of all input modules present in the local CPU base and local expansion bases. The status of each input is stored in the input image register area. The input image register data is used by the CPU when it solves the application program.
You may be thinking, "What if an input changes after the CPU has read the inputs?" Good question! Generally, the CPU program execution time is measured in milliseconds, so in most cases this is not a problem. If you need to know more, I/O response timing is explained in more detail later in this chapter.
The CPU also uses the output image register to update the output modules present in the local CPU base and local expansion bases.


Service Peripherals After the CPU updates the I/O points, it reads any attached peripheral devices. This is primarily a communications service for any attached devices. For example, if you were using an operator interface to read or write data, the CPU would service these requests during this portion of the scan.


Service for
Monitoring and
Forcing Operations

After the CPU updates any communications requests from peripheral devices, it determines if any forcing operations have been requested. The CPU also services any monitoring requests during this portion. For example, if you are using the Handheld Programmer to monitor the current value of a timer or counter, the CPU will provide this information during this portion of the scan.
Here's an example of one of the more popular requests. For example, you may want to force an input on, even though it is really off. This allows you to change the point status that was stored in the image register. This value will be valid until the image register location is written to during the Update I/O Points segment of the next scan.


It is important to note the DL305 CPUs only retain the forced value for one scan if the input point used corresponds to a module that is installed in the base. The following example shows how the forcing actually works on the next CPU scan.


As you can see from the example, the input forcing will not be valid when the CPU reads the input status on the next scan.
Output point forcing works in a similar manner. That is, if you force an output on and the application program results dictate the output should be turned on, then the output image register will show the results of the application program instead of the forcing request. This is discussed in more detail in the next section.

Solve Application The CPU evaluates each instruction in the application program during this segment Program

System Operation
NOTE: If an output point was used in the application program, the results of the program solution will overwrite any forcing information that was stored. For example, if output 030 was forced on by the programming device, and a rung containing 030 was evaluated such that 030 should be turned off, then the output image register will show that 030 should be off. Of course, you can force output points that are not used in the application program. In this case, the point remains forced because there is no solution that results from the application program execution.


## I/O Response Time

Is Timing Important I/O response time is the amount of time required for the control system to sense a
for Your Application? change in an input point and update a corresponding output point. In the majority of applications, the CPU performs this task in such a short period of time you may never have to concern yourself with the aspects of system timing. However, some applications do require extremely fast update times. In these cases, you may need to know how to to determine the amount of time spent during the various segments of operation.
There are four things that can affect the I/O response time.

- The point in the cycle when the field input changes states
- Input module Off to On delay time
- CPU scan time
- Output module Off to On delay time

The next paragraphs show how these items interact to affect the response time.
Normal Minimum
I/O Response
I/O Response

The I/O response time is shortest when the module senses the input change just before the I/O Update portion of the execution cycle. In this case the input status is read, the application program is solved, and the output point gets updated on the following scan. The following diagram shows an example of the timing for this situation.


In this case, you can calculate the response time by simply adding the following items.

Input Delay + Scan Time + Output Delay = Response Time

Normal Maximum The I/O response time is longest when the module senses the input change just after
I/O Response the Update I/O portion of the execution cycle. In this case the new input status does not get read until the following scan. The following diagram shows an example of the timing for this situation.


In this case, you can calculate the response time by simply adding the following items.
Input Delay $+(2 \times$ Scan Time $)+$ Output Delay $=$ Response Time

Improving Response Time

There are a few things you can do the help improve throughput.

- You can try to choose instructions with faster execution times
- You can choose modules that have faster response times


## CPU Scan Time Considerations

The scan time covers all the cyclical
tasks that are performed by the operating system. This information can be very important when evaluating the performance of a system.
As we've shown previously there are several segments that make up the overall execution cycle. Each of these segments requires a certain amount of time to complete. Of all the segments, the only ones you really need to understand are those that occur during Run Mode. Even within this portion, your primary concern should be to understand the instruction execution times.


The following table provides execution timing guidelines for the execution cycle.

D A timer interrupt occurs every 2.5 ms during the scan. The interrupt requires approximately $266 \mu \mathrm{~s}$ to service. So, the larger the program, the more timer interrupts.


DL330 / DL330P Scan Calculation
$\mathbf{A}=3.3 \mathrm{~ms}$ typical $\mathrm{I} / \mathrm{O}$ update
$\mathbf{B}=0-5.2 \mathrm{~ms}$ maximum to service peripherals, monitoring, and forcing
$\mathbf{C}=$ Total of instruction execution time
$\mathbf{D}=266 \mu s \times \frac{A+B+C+D}{2.5 m s}$
Actual Scan $=A+B+C+D$

NOTE: There are other events that occur during the execution cycle, but the areas shown are the most important. This information is provided so you will understand the basic scan calculations. Scan time can vary from scan-to-scan.

DL340 Scan Calculation

Typical scan overhead is from $2.5-3.5 \mathrm{~ms}$. However, the following table provides more precise execution timing guidelines for the execution cycle.


D A timer interrupt occurs every 2.5 ms during the scan. The interrupt requires approximately $200 \mu \mathrm{~s}$ to service. So, the larger the program, the more timer interrupts.

A = 2 ms typical $\mathrm{I} / \mathrm{O}$ update
$B=0-1.2$ ms maximum to service peripherals, monitoring, and forcing
$\mathbf{C}=$ Total of instruction execution time
$\mathbf{D}=200 \mu s \times \frac{A+B+C+D}{2.5 m s}$
Actual Scan $=A+B+C+D$

NOTE: There are other events that occur during the execution cycle, but the areas shown are the most important. This information is provided so you will understand the basic scan calculations. Scan time can vary from scan-to-scan.

Application Program Execution

The CPU processes the program from address 0 to the END instruction. The CPU executes the program left to right and top to bottom. As each rung is evaluated the appropriate image register or memory location is updated.
The time required to solve the application program depends on the type and number of instructions used
You can add the execution times for all the instructions in your program to determine how much time is required to execute the instructions.
For example, the execution time for a DL330 running the program shown would be calculated as follows.

| Instruction | Time |
| :--- | ---: |
| STR 000 | $6.6 \mu \mathrm{~s}$ |
| OR 160 | $6.6 \mu \mathrm{~s}$ |
| ANDN 001 | $8.4 \mu \mathrm{~s}$ |
| OUT 020 | $7.5 \mu \mathrm{~s}$ |
| STRN 161 | $9.1 \mu \mathrm{~s}$ |
| DSTR K10 | $14.3 \mu \mathrm{~s}$ |
| STRN 162 | $9.1 \mu \mathrm{~s}$ |
| DOUT R400 | $52.6 \mu \mathrm{~s}$ |
| STRN 163 | $9.1 \mu \mathrm{~s}$ |
| DSTR K50 | $14.3 \mu \mathrm{~s}$ |
| STRN 164 | $9.1 \mu \mathrm{~s}$ |
| DOUT R402 | $52.6 \mu \mathrm{~s}$ |
| STR 005 | $6.6 \mu \mathrm{~s}$ |
| ANDN 010 | $8.4 \mu \mathrm{~s}$ |
| OUT 021 | $7.5 \mu \mathrm{~s}$ |
| END | $\sim 0 \mu \mathrm{~s}$ |
| TOTAL | $221.8 \mu \mathrm{~s}$ |

NOTE: Appendix C provides the instruction execution times for the DL305 CPUs.

## Memory Map



Octal Numbering System

With any PLC system, you generally have many different types of information to process. This includes input device status, output device status, various timing elements, parts counts, etc. It is important to understand how the system represents and stores the various types of data. For example, you need to know how the system identifies input points, output points, data words, etc. The following paragraphs discuss the various memory types used in the DL305 CPUs.

NOTE: The DL305 CPUs do not all have the same memory ranges. Make sure you review the detailed memory maps at the end of this section to determine the available memory types for your particular model of CPU.

All memory locations or areas are numbered in Octal (base 8). For example, the diagram shows how the octal numbering system works for the discrete input points. Notice the octal system does not contain any numbers with the digits 8 or 9 .


Two Basic Memory Types: Discrete and Word

As you examine the different memory types, you'll notice two types of memory in the DL305, discrete and word memory. Discrete memory is one bit that can be either a 1 or a 0 . Word memory is referred to as Data Register memory and is an 8 -bit location normally used to manipulate data/numbers, store data/numbers, etc.
Some information is automatically stored in Register (R) memory. For example, the timer current values are stored in Registers that correspond to the timer or counter number. So, the current value for timer T600 is automatically stored in R600.

Discrete - On or Off, 1 bit


Word Locations - 8 bits
00110 d 101

## R Memory Locations for Discrete Memory Areas

The discrete memory area is for inputs, outputs, control relays, etc. However, you can also access the bit data types as an R-memory word. Each R-memory location contains 8 consecutive discrete locations.
Remember, the DL305 system does not have a separate memory type for input and output points. The type of point assigned to the location depends on the type of module installed in the slot that corresponds to the register location. Also, the number of registers assigned to the module depends on the number of points. For example, a 16-point module would require two registers since each register only contains 8 bits. The following diagram shows how the points for a 16-point module installed in the slot next to the CPU would map into registers.

16 Discrete Input Module


These discrete memory areas and their corresponding R-memory ranges are listed in the memory area tables at the end of this chapter.

The discrete input and output points do not have separate data types. The type of point assigned to the reference address depends on the type of module installed in the base. Depending on the type of CPU, you can have up to 184 I/O points in a DL305 system.

In the first example, output point 010 will be turned on when input 000 energizes. This assumes an 8-point input module is installed in the first slot and an 8-point output module is installed in the second slot.


1st Slot - Output, 2nd Slot - Input


NOTE: Unused I/O references can be used as control relays in the application program.

## Control Relays

Control relays are discrete bits normally used to control the user program. The control relays do not represent a real world device, that is, they cannot be physically tied to switches, output coils, etc. They are internal to the CPU. Because of this, control relays can be programmed as discrete inputs or discrete outputs. These locations are used in programming the discrete memory locations (C) or the corresponding word location which contains 8 consecutive discrete locations.

In this example, memory location 160 will energize when input 010 turns on. The second rung shows a simple example of how to use a control relay as an input.

NOTE: Some of the references normally assigned as Control Relays can also be used to refer to a16-point I/O modules in some situations. Make sure you review the memory maps at the end of this chapter if you use 16 -point modules.

Timers and
Timer Status Bits (T Data type)

You can have up to 64 timers/counters in a DL305 CPU. Both the timers and counters share the same memory area. This means you cannot have a timer T600 and a counter CT600 in the same program.
When you use these locations for timers, each timer has a status bit that reflects the relationship between the current value and the timer preset value. The timer status bit will be on when the current value is equal or greater than the preset value of a corresponding timer. The DL330P does not support the status bit. Instead, you have to use comparative boolean contacts. See Chapter 12 for details.
In the example shown, input 000 turns on to start timer T600. When the timer reaches the preset of 3 seconds (K of 30) timer status contact T600 turns on. When contact T600 turns on, output 022 is energized.


You can have up to 64 timers/counters in a DL305 CPU. Both the timers and counters share the same memory area. This means you cannot have a timer T600 and a counter CT600 in the same program.
When you use these locations for counters, each counter has a status bit that reflects the relationship between the current count and the preset value. The counter status bit will be on when the current value is equal to or greater than the counter preset value. The DL330P does not support the status bit. Instead, you have to use comparative boolean contacts. See Chapter 12 for details.
In the example shown, Each time contact 000 transitions from off to on, the counter increments by one. (If 001 comes on, the counter is reset to zero.) When the counter reaches the preset of 10 counts ( K of 10) counter status contact CT603 turns on. When CT603 turns on, output 022 turns on.


As mentioned earlier, some information is automatically stored in R memory. This is true for the current values associated with counters. For example, R600 holds the current value for counter 600, R601 holds the current value for counter 601, etc.
The primary reason for this is programming flexibility. The example shows how you can use comparative contacts to monitor several count values from a single counter.


Data Registers (R Data Type)

A word memory location is referred to as a Data Register, which is an 8-bit location normally used to manipulate data/numbers, store data/numbers, etc. Some information is automatically stored in registers. For example, the timer current values are automatically stored in a register that corresponds to the timer or counter number being used.
The example shows how a four-digit BCD constant is loaded into the accumulator and then stored in a Register location. Notice two registers are required to hold the 4-digit number.

Stages are used in RLL PLUS programs to create a structured program, similar to a flowchart. Each program stage denotes a program segment. When the program segment or stage, is active, the logic within that segment is executed. If the stage is off or inactive, the logic is not executed and the CPU skips to the next active stage. (See Chapter 10 for a more detailed description of RLLPLUS programming.)
Each stage also has a discrete status bit that can be used as an input to indicate whether the stage is active or inactive. If the stage is active, then the status bit is on. If the stage is inactive, then the status bit is off. This status bit can also be turned on or off by other instructions, such as the SET or RESET instructions. This allows you to easily control stages throughout the program.


Ladder Representation


Shift Registers

System Operation
Special Relays Special relays are discrete memory locations with pre-defined functionality. There are many different types of special relays. For example, some aid in programming, others provide system operating status information, etc.
In this example, special relay 375 will energize for 50 ms and de-energize for energize for 50 ms and de-energize for
50 ms because relay 375 is a pre-defined relay that will be on for 50 ms and off for 50 ms .

Special Registers (R Data Type) completely separate areas and are not related. bits in each one, then you could have up to 16 Shift Registers.
In the example shown, contact 001 represents the data value ( 0 or 1 ) that will be loaded into the shift register when the clock input (002) is active. Each time the clock input comes on, the data values are shifted through the bit positions from 400 to 417 . Input 003 resets the shift register and sets all the bit positions back to zero. Chapter 11 provides detailed instructions on how to use shift registers.

There are 128 bits available for use in Shift Registers with the DL330 and DL340 CPUs. You can still use Shift Registers in the DL330P CPU, but a separate range of bits is not provided. You have to use the control relay points in the Shift Register instructions. These are numbered from 400 to 577 . Your first reaction may be to think these are somehow related to the Data Registers with the same numbers. They are

The number of Shift Register instructions that can be used depends on how many bits are used with each Shift Register. For example, if you have a DL330 and you use 16 bits in each Shift Register, you can have up to 8 Shift Registers. If you only used 8

There are also a few special registers that store various types of system information. For example, the DL340 communication port parameters are set in special registers. The detailed memory maps at the end of this chapter show special register assignments for each CPU.

## DL305 Aliases

An alias is an alternate way of referring to certain memory types, such as timer/counter current values, register locations for I/O points, etc.. This makes some instructions easier to understand. The use of the alias is optional, but some users may find the alias to be helpful when developing a program. The table below shows how the aliases can be used.

| Address Start | Alias Start | Example |
| :---: | :---: | :--- |
| R0 | RIO0 | R0 is the 8-bit word memory reference for discrete bits IO0 <br> through IO7, hence its alias is RIO0. R1 is the 8-bit word <br> memory reference for discrete bits IO10 through IO17, hence <br> its alias is RIO10. |
| R16 | RC160 | R16 is the 8-bit word memory reference for control relays <br> C160 through C167, hence its alias is RC160. R17 is the 8-bit <br> word memory reference for control relays C170 through C177, <br> hence its alias is RC170. |
| R40 | RS400 | R40 is the 8-bit word memory reference for stage bits S400 <br> through S407, hence its alias is RS400. R41 is the 8-bit word <br> memory reference for stage bits S410 through S417, hence its <br> alias is RS410. |
| R70 | RIO700 | R70 is the 8-bit word memory reference for discrete bits IO700 <br> through IO707, hence its alias is RIO700. R71 is the 8-bit word <br> memory reference for discrete bits IO710 through IO717, <br> hence its alias is RIO710. |
| R77 | RC770 | R77 is the 8-bit word memory reference for control relays <br> C770 through C777, hence its alias is RC770. |
| R100 | RC1000 | R100 is the 8-bit word memory reference for control relays <br> C1000 through C1007, hence its alias is RC1000. R101 is the <br> 8-bit word memory reference for control relays C1010 through <br> C1017, hence its alias is RC1010. |
| R107 | RC1070 | R107 is the 8-bit word memory reference for control relays <br> C1070 through C1077, hence its alias is RC1070. |
| R600 | TCA600 | R600 is the 16-bit timer/counter accumulator value for <br> timer/counter 600, hence its alias is TCA600. R601 is the <br> $16-$ bit timer/counter accumulator value for timer/counter 601, <br> hence its alias is TCA601. |

System Operation

## DL330 Memory <br> Map

| Memory Type | Discrete Memory Reference (octal) | Register Memory Reference (octal) | Qty. Decimal | Symbol |
| :---: | :---: | :---: | :---: | :---: |
| Input / Output Points | $\begin{aligned} & 000-157 \\ & 700-767 \end{aligned}$ | $\begin{aligned} & \text { R000-R015 } \\ & \text { R070-R076 } \end{aligned}$ | 168 Total | $\begin{array}{cc} 10000 \\ +\quad-\quad-\quad \end{array}$ |
| Control Relays | 160-373 | R016-R037 | 140 |  |
| Special Relays | $\begin{aligned} & 374-377 \\ & 770-777 \end{aligned}$ | $\begin{array}{\|l} \text { R037 } \\ \text { R077 } \end{array}$ | 12 | $\stackrel{c}{\text { C772 }} \underset{-}{\text { C376 }}$ |
| Timers / Counters | $\begin{aligned} & 600-673 \\ & 674-677^{*} \end{aligned}$ | None | 64 | $\underbrace{\mathrm{K}_{100}^{\text {TMR }}{ }^{\text {T600 }}}-\begin{gathered} \text { CNT CT600 } \\ \text { K10 } \end{gathered}$ |
| Timer / Counter Current Values | None | $\begin{aligned} & \text { R600 - R673 } \\ & \text { R674 - R677* } \end{aligned}$ | 64 | Contact valid for counters only. |
| Timer / Counter Status Bits | $\begin{aligned} & \hline \text { T600 - T673 } \\ & \text { T674 - T677 } \end{aligned}$ | None | 64 | $\begin{gathered} \mathrm{T} 600 \\ -\mid \vdash \end{gathered}$ |
| Data Words | None | R400-R563 | 116 | None specific, used with many instructions |
| Shift Registers | 400-577 | None | 128 |  |
| Special Registers | None | R574-R577 | 4 | R574 - R575 used with FAULT <br> R576 - R577 Auxiliary Accumulator |

[^0]
## DL330P Memory <br> Map

| Memory Type | Discrete Memory Reference (octal) | Register Memory Reference (octal) | Qty. Decimal | Symbol |
| :---: | :---: | :---: | :---: | :---: |
| Input / Output Points | $\begin{aligned} & 000-157 \\ & 700-767 \end{aligned}$ | $\begin{array}{\|l\|l} \text { R000 - R015 } \\ \text { R070 - R076 } \end{array}$ | 168 Total | $\begin{array}{ll} \hline 10000 & 10010 \\ -H & -() \end{array}$ |
| Control Relays | $\begin{array}{\|l\|} \hline 160-174 \\ 200-277 \end{array}$ | $\begin{aligned} & \text { R016 - R017 } \\ & \text { R020 - R027 } \end{aligned}$ | 77 |  |
| Special Relays | $\begin{aligned} & 175-177 \\ & 770-777 \end{aligned}$ | $\begin{array}{\|l\|l\|} \hline \text { R017 } \\ \text { R077 } \end{array}$ | 11 | $\begin{array}{cc} \mathrm{C} 772 \\ -\quad-\quad-\quad)^{\mathrm{C} 376} \end{array}$ |
| Timers / Counters | $\begin{aligned} & 600-673 \\ & 674-677^{*} \end{aligned}$ | None | 64 | $-\underbrace{\text { TMR }_{\text {K100 }}{ }^{\text {T600 }}}-$KNT CT600 |
| Timer / Counter Current Values | None | $\begin{aligned} & \text { R600 - R673 } \\ & \text { R674 - R677* } \end{aligned}$ | 64 | $\stackrel{\text { CTA600 }}{ }\left\|\geq\left.\right\|^{\text {K250 }} \quad \xrightarrow{\text { TA600 }}\right\| \geq\left.\right\|^{\text {K250 }}$ |
| Timer / Counter Status Bits | $\begin{aligned} & \text { T600 - T673 } \\ & \text { T674 - T677* } \end{aligned}$ | None | 64 | $\begin{aligned} & \text { T600 } \\ & -\mid \quad \end{aligned}$ |
| Data Words | None | R400-R563 | 116 | None specific, used with many instructions |
| Stages | S0-S177 | R100-R117 | 128 |  |
| Special Registers | None | R574-R577 | 4 | R574-R575 used with FAULT <br> R576 - R577 Auxiliary Accumulator |

* T/ C Setpoint Unit Only. Can be used as data registers if the Timer/Counter Setpoint Unit or Thumbwheel Interface Module is not used, which provides a total of 128 data registers.
R564 - R573 contain the preset value used with the Timer / Counter Setpoint Unit. R674 - R677 contain the current values for these timers or counters

Timer / Counter Current Values Registers (T600-T677) are 16-bit registers.

System Operation

## DL340 Memory <br> Map

| Memory Type | Discrete Memory Reference (octal) | Register Memory Reference (octal) | Qty. Decimal | Symbol |
| :---: | :---: | :---: | :---: | :---: |
| Input / Output Points | $\begin{aligned} & 000-157 \\ & 700-767 \end{aligned}$ | $\begin{aligned} & \text { R000 - R015 } \\ & \text { R070 - R076 } \end{aligned}$ | 168 Total | $\begin{array}{cc} 10000 \\ +\quad-\quad-() \end{array}$ |
| Control Relays | $\begin{aligned} & 160-373 \\ & 1000-1067 \end{aligned}$ | $\begin{array}{\|l\|l} \text { R016 - R037 } \\ \text { R100 - R106 } \end{array}$ | 180 |  |
| Special Relays | $\left\lvert\, \begin{aligned} & 374-377 \\ & 770-777 \\ & 1070-1077 \end{aligned}\right.$ | $\begin{array}{\|l} \text { R037 } \\ \text { R077 } \\ \text { R107 } \end{array}$ | 20 | $\begin{array}{cc} \mathrm{C} 772 \\ -\quad=- & \mathrm{C}^{\mathrm{C} 376} \end{array}$ |
| Timers / Counters | $\begin{aligned} & 600-673 \\ & 674-677^{*} \end{aligned}$ | None | 64 | $-\underbrace{-{ }^{\text {TMR }}{ }^{\text {K100 }}{ }^{\text {T600 }}}-$CNT CT600 |
| Timer / Counter Current Values | None | $\begin{array}{\|l} \text { R600 - R673 } \\ \text { R674 - R677* } \end{array}$ | 64 | $\stackrel{\text { CTA600 }}{=}={ }^{\text {K250 }}$ <br> Contact valid for counters only. |
| Timer / Counter Status Bits | $\begin{aligned} & \text { T600 - T673 } \\ & \text { T674-T677* } \end{aligned}$ | None | 64 | $\begin{gathered} \mathrm{T} 600 \\ -\mid \end{gathered}$ |
| Data Words | None | $\begin{aligned} & \text { R400 - R563 } \\ & \text { R700 - R767 } \end{aligned}$ | 172 | None specific, used with many instructions |
| Shift Registers | 400-577 | None | 128 | $\left.-\begin{array}{c}\text { SR } \\ 400 \\ 417 \\ \hline\end{array}\right]$ |
| Special Registers | None | $\begin{array}{\|l} \text { R574 - R577 } \\ \text { R770 - R777 } \end{array}$ | 12 | R574-R575 used with FAULT R576-R577 Auxiliary Accumulator R770-R777 Communications Setup |

*T/ C Setpoint Unit Only. Can be used as data registers if the Timer/Counter Setpoint Unit or Thumbwheel Interface Module is not used. R564-R573 contain the preset value used with the Timer / Counter Setpoint Unit. R674-R677 contain the current values for these timers or counters.

Registers T600-T677 are 16-bit registers.

## I/O Point Bit Map

These tables provide a listing of the individual Input points associated with each register location for the DL330, DL330P, and DL340 CPUs.

| MSB | I/O References |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 007 | 006 | 005 | 004 | 003 | 002 | 001 | 000 | R0 |
| 017 | 016 | 015 | 014 | 013 | 012 | 011 | 010 | R 1 |
| 027 | 026 | 025 | 024 | 023 | 022 | 021 | 020 | R 2 |
| 037 | 036 | 035 | 034 | 033 | 032 | 031 | 030 | R 3 |
| 047 | 046 | 045 | 044 | 043 | 042 | 041 | 040 | R 4 |
| 057 | 056 | 055 | 054 | 053 | 052 | 051 | 050 | R 5 |
| 067 | 066 | 065 | 064 | 063 | 062 | 061 | 060 | $\underline{R 6}$ |
| 077 | 076 | 075 | 074 | 073 | 072 | 071 | 070 | R 7 |
| 107 | 106 | 105 | 104 | 103 | 102 | 101 | 100 | R 10 |
| 117 | 116 | 115 | 114 | 113 | 112 | 111 | 110 | R 11 |
| 127 | 126 | 125 | 124 | 123 | 122 | 121 | 120 | R 12 |
| 137 | 136 | 135 | 134 | 133 | 132 | 131 | 130 | R 13 |
| 147 | 146 | 145 | 144 | 143 | 142 | 141 | 140 | R 14 |
| 157 | 156 | 155 | 154 | 153 | 152 | 151 | 150 | R 15 |
| 167 | 166 | 165 | 164 | 163 | 162 | 161 | 160 | $\mathrm{n} / \mathrm{a}$ |
| 177 | 176 | 175 | 174 | 173 | 172 | 171 | 170 | $\mathrm{n} / \mathrm{a}$ |
| 707 | 706 | 705 | 704 | 703 | 702 | 701 | 700 | R 70 |
| 717 | 716 | 715 | 714 | 713 | 712 | 711 | 710 | R 71 |
| 727 | 726 | 725 | 724 | 723 | 722 | 721 | 720 | R 72 |
| 737 | 736 | 735 | 734 | 733 | 732 | 731 | 730 | R 73 |
| 747 | 746 | 745 | 744 | 743 | 742 | 741 | 740 | R 74 |
| 757 | 756 | 755 | 754 | 753 | 752 | 751 | 750 | R 75 |
| 767 | 766 | 765 | 764 | 763 | 762 | 761 | 760 | R 76 |



NOTE: 160-167 can be used as I/O in a DL330 or DL330P CPU under certain conditions. 160-177 can be used as I/O in a DL340 CPU under certain conditions. You should consult Chapter 4 to determine which configurations allow the use of these points.

These points may be used as control relays. You cannot use them as both control relays and as I/O points. Also, if you use these points as I/O, you cannot access these I/O points as a Data Register reference using the DSTR5 (F55) and DOUT5 (F65) functions.

## Control Relay Bit Map

The following tables provide a listing of the individual control relays associated with each register location for the DL305 CPUs.

NOTE: If a 16 pt module is used in Slot 6 for the DL330 or DL330P CPU, 160 through 167 will not be available for control relay assignments. If a 16pt module is used in Slot 6 and/or Slot 7 for a DL340 CPU, 160-167 and/or 170-177 are not available for control relay assignments. You cannot use these points as both control relays and as I/O points. If you use these points as I/O points, you still enter them as C160-C177 in DirectSOFT.

Also, if you use these points as I/O, you cannot access these I/O points as a Data Register reference using the DSTR5 (F55) and DOUT5 (F65) functions.

$\left.$| MSB | DL330 |  |  |  |  |  |  | Control Relay References |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | | Register |
| :--- |
| Number | \right\rvert\,

[^1]$\left.$| MSB | DL330P <br> Control Relay References |  |  |  |  |  |  | LSB |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | | Register |
| :--- |
| Number | \right\rvert\,

* Control relays 200-277 can be made retentive by setting a CPU dipswitch. See Chapter 3 for details on setting CPU dipswitches.

| MSB | DL340 |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 167 | 166 | 165 | 164 | 163 | 162 | 161 | 160 | R16 |
| 177 | 176 | 175 | 174 | 173 | 172 | 171 | 170 | R17 |
| 207 | 206 | 205 | 204 | 203 | 202 | 201 | 200 | R20 |
| 217 | 216 | 215 | 214 | 213 | 212 | 211 | 210 | R21 |
| 227 | 226 | 225 | 224 | 223 | 222 | 221 | 220 | R22 |
| 237 | 236 | 235 | 234 | 233 | 232 | 231 | 230 | R23 |
| 247 | 246 | 245 | 244 | 243 | 242 | 241 | 240 | R24 |
| 257 | 256 | 255 | 254 | 253 | 252 | 251 | 250 | R25 |
| 267 | 266 | 265 | 264 | 263 | 262 | 261 | 260 | R26 |
| 277 | 276 | 275 | 274 | 273 | 272 | 271 | 270 | R27 |
| 307 | 306 | 305 | 304 | 303 | 302 | 301 | 300 | R30 |
| 317 | 316 | 315 | 314 | 313 | 312 | 311 | 310 | R31 |
| 327 | 326 | 325 | 324 | 323 | 322 | 321 | 320 | R32 |
| 337 | 336 | 335 | 334 | 333 | 332 | 331 | 330 | R33 |
| 347 | 346 | 345 | 344 | 343 | 342 | 341 | $340^{*}$ | R34 |
| 357 | 356 | 355 | 354 | 353 | 352 | 351 | 350 | R35 |
| 367 | 366 | 365 | 364 | 363 | 362 | 361 | 360 | R36 |
|  |  |  |  |  |  |  |  |  |
| 1007 | 1006 | 1005 | 1004 | 1003 | 1002 | 1001 | 1000 | R100 |
| 1017 | 1016 | 1015 | 1014 | 1013 | 1012 | 1011 | 1010 | R101 |
| 1027 | 1026 | 1025 | 1024 | 1023 | 1022 | 1021 | 1020 | R102 |
| 1037 | 1036 | 1035 | 1034 | 1033 | 1032 | 1031 | 1030 | R103 |
| 1047 | 1046 | 1045 | 1044 | 1043 | 1042 | 1041 | 1040 | R104 |
| 1057 | 1056 | 1055 | 1054 | 1053 | 1052 | 1051 | 1050 | R105 |
| 1067 | 1066 | 1065 | 1064 | 1063 | 1062 | 1061 | 1060 | R106 |

[^2]System Operation

## Special Relays

The following table shows the Special Relays used with the DL305 CPUs. Note, our DL105, DL205, and DL405 product families use the data type "SP" to designate Special Relays. Even though we refer to the following relays as special relays, DirectSOFT uses the letter "C" as a special relay prefix for the DL305 products. These letters aren't used with the handheld programmer.

| CPUs | Special Relay | Description of Contents |
| :---: | :---: | :---: |
| DL330P | C175 | 100 ms clock , on for 50 ms and off for 50 ms . |
|  | C176 | Disables all outputs except for those entered with the SET OUT instruction. |
|  | C177 | Battery voltage is low. |
| $\begin{aligned} & \text { DL330 } \\ & \text { DL340 } \end{aligned}$ | C374 | On for the first scan cycle after the CPU is switched to Run Mode. |
|  | C375 | 100 ms clock, on for 50 ms and off for 50 ms . |
|  | C376 | Disables all outputs except for those entered with the SET OUT instruction. |
|  | C377 | Battery voltage is low. |
| $\begin{gathered} \text { DL330 } \\ \text { DL330P } \\ \text { DL340 } \end{gathered}$ | C770 | Changes timers to 0.01 second intervals. Timers are normally 0.1 second time intervals. |
|  | C771 | The external diagnostics FAULT instruction (F20) is in use. |
|  | C772 | The data in the accumulator is greater than the comparison value. |
|  | C773 | The data in the accumulator is equal to the comparison value. |
|  | C774 | The data in the accumulator is less than the comparison value. |
|  | C775 | An accumulator carry or borrow condition has occurred. |
|  | C776 | The accumulator value is zero. |
|  | C777 | The accumulator has an overflow condition. |
| DL340 | C1072 | Port 2 parity: on = odd, off = none |
|  | C1074 | The RX or WX instruction is active. |
|  | C1075 | An error occurred during communications with the RX or WX instructions. |
|  | C1076 | Port 2 communications mode: on = ASCII mode, off = HEX mode. DirectNET supports both ASCII and HEX modes and Modbus ${ }^{\circledR}$ only supports HEX mode. |
|  | C1077 | Port 1 communications mode: on = ASCII mode, off = HEX mode |

## Timer / Counter Registers and Contacts

The following table shows the locations used for programming timer or counters. Since timers and counters share the same data area, you cannot have timers and counters with duplicate numbers. For example, if you have Timer 600, you cannot have a Counter 600.
Each register contains the current value for the timer or counter. Each timer or counter also has a timer or counter contact with the same reference number.

NOTE: Counter current values are retentive and retain their state after a power cycle. These registers are 16-bit registers.

| Timer/Counter References/Registers |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 607 | 606 | 605 | 604 | 603 | 602 | 601 | 600 |
| 617 | 616 | 615 | 614 | 613 | 612 | 611 | 610 |
| 627 | 626 | 625 | 624 | 623 | 622 | 621 | 620 |
| 637 | 636 | 635 | 634 | 633 | 632 | 631 | 630 |
| 647 | 646 | 645 | 644 | 643 | 642 | 641 | 640 |
| 657 | 656 | 655 | 654 | 653 | 652 | 651 | 650 |
| 667 | 666 | 665 | 664 | 663 | 662 | 661 | 660 |
| $677^{\star}$ | $676^{\star}$ | $675^{\star}$ | $674^{\star}$ | 673 | 672 | 671 | 670 |

* Used with Timer / Counter Setpoint Unit and /or Thumbwheel Interface Module.

Registers 674-677 are used in programming for use with the Timer/Counter Setpoint Unit and the Thumbwheel Interface Module that are available in some compatible product families. The registers contain the current time or count. There is also a status bit for each register with the same reference number. For example, the current value for Timer 674 is stored in R674 and the status contact is T674.

The presets for these modules are stored in R564-R573 as follows.

- R564 - R565 - 1st T/C preset
- R566 - R567 - 2nd T/C preset
- R570 - R571 - 3rd T/C preset
- R572 - R573 - 4th T/C preset


The example shows how a 4-digit number would be represented in these registers.

## Data Registers

The following 8-bit data registers are primarily used with data instructions to store various types of application data. For example, you could use a register to hold a timer or counter preset value.
Some data instructions call for two bytes, which will correspond to two consecutive 8 -bit data registers such as R401 and R400. The LSB (Least Significant Bit) will be in register R400 as bit0 and the MSB (Most Significant Bit) will be in register R401 as bit17.

NOTE: Data Registers are retentive.

| DL330 / DL330P <br> 8-Bit Data Registers |  |  |  |  |  |  |  |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| 407 | 406 | 405 | 404 | 403 | 402 | 401 | 400 |
| 417 | 416 | 415 | 414 | 413 | 412 | 411 | 410 |
| 427 | 426 | 425 | 424 | 423 | 422 | 421 | 420 |
| 437 | 436 | 435 | 434 | 433 | 432 | 431 | 430 |
| 447 | 446 | 445 | 444 | 443 | 442 | 441 | 440 |
| 457 | 456 | 455 | 454 | 453 | 452 | 451 | 450 |
| 467 | 466 | 465 | 464 | 463 | 462 | 461 | 460 |
| 477 | 476 | 475 | 474 | 473 | 472 | 471 | 470 |
| 507 | 506 | 505 | 504 | 503 | 502 | 501 | 500 |
| 517 | 516 | 515 | 514 | 513 | 512 | 511 | 510 |
| 527 | 526 | 525 | 524 | 523 | 522 | 521 | 520 |
| 537 | 536 | 535 | 534 | 533 | 532 | 531 | 530 |
| 547 | 546 | 545 | 544 | 543 | 542 | 541 | 540 |
| 557 | 556 | 555 | 554 | 553 | 552 | 551 | 550 |
|  |  |  |  |  |  |  |  |


| DL340 <br> 8-Bit Data Registers |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 407 | 406 | 405 | 404 | 403 | 402 | 401 | 400 |  |  |  |  |  |  |  |
| 417 | 416 | 415 | 414 | 413 | 412 | 411 | 410 |  |  |  |  |  |  |  |
| 427 | 426 | 425 | 424 | 423 | 422 | 421 | 420 |  |  |  |  |  |  |  |
| 437 | 436 | 435 | 434 | 433 | 432 | 431 | 430 |  |  |  |  |  |  |  |
| 447 | 446 | 445 | 444 | 443 | 442 | 441 | 440 |  |  |  |  |  |  |  |
| 457 | 456 | 455 | 454 | 453 | 452 | 451 | 450 |  |  |  |  |  |  |  |
| 467 | 466 | 465 | 464 | 463 | 462 | 461 | 460 |  |  |  |  |  |  |  |
| 477 | 476 | 475 | 474 | 473 | 472 | 471 | 470 |  |  |  |  |  |  |  |
| 507 | 506 | 505 | 504 | 503 | 502 | 501 | 500 |  |  |  |  |  |  |  |
| 517 | 516 | 515 | 514 | 513 | 512 | 511 | 510 |  |  |  |  |  |  |  |
| 527 | 526 | 525 | 524 | 523 | 522 | 521 | 520 |  |  |  |  |  |  |  |
| 537 | 536 | 535 | 534 | 533 | 532 | 531 | 530 |  |  |  |  |  |  |  |
| 547 | 546 | 545 | 544 | 543 | 542 | 541 | 540 |  |  |  |  |  |  |  |
| 557 | 556 | 555 | 554 | 553 | 552 | 551 | 550 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | 7 |  | 563 | 562 | 561 | 560 |
| 707 | 706 | 705 | 704 | 703 | 702 | 701 | 700 |  |  |  |  |  |  |  |
| 717 | 716 | 715 | 714 | 713 | 712 | 711 | 710 |  |  |  |  |  |  |  |
| 727 | 726 | 725 | 724 | 723 | 722 | 721 | 720 |  |  |  |  |  |  |  |
| 737 | 736 | 735 | 734 | 733 | 732 | 731 | 730 |  |  |  |  |  |  |  |
| 747 | 746 | 745 | 744 | 743 | 742 | 741 | 740 |  |  |  |  |  |  |  |
| 757 | 756 | 755 | 754 | 753 | 752 | 751 | 750 |  |  |  |  |  |  |  |
| 767 | 766 | 765 | 764 | 763 | 762 | 761 | 760 |  |  |  |  |  |  |  |

## Stage Control / Status Bit Map

This table provides a listing of the individual stages and stage control bits. These are only available with the DL330P CPU.

$\left.$| MSB | Stage References |  |  |  |  |  |  | LSB |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | | Register |
| :--- |
| Number | \right\rvert\,

## Shift Register Bit Map

The shift register bits listed below are used in the shift register instruction. These outputs are discrete bits and are not the same locations as the 8 Bit Data Registers. These bits are retentive meaning they retain their state after a power cycle.


NOTE: The DL330P does not have Shift Register bits. Shift Register instructions in the DL330P use Control Relays memory references.

| MSB | DL330 / DL340 |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Shift Register References |  |  |  |  |  |  |  |$\quad$ LSB | Register |
| :---: |
| Number |$|$

With the DL340 CPU, these bits can also be used as control relays if they are not used with a Shift Register instruction.

System Operation

## Special Registers

This table provides a listing of the special registers used with the DL305 CPUs.

| CPUs | Special Register | Description of Contents |
| :---: | :---: | :---: |
| DL330 | R574-575 | Contains the error code used with the FAULT instruction. |
| $\begin{gathered} \text { DL330P } \\ \text { DL340 } \end{gathered}$ | R576-577 | Auxiliary accumulator used with the MUL and DIV instructions. |
| DL340 Only | R771 | Sets the upper byte of the station address assigned to the bottom communication port. Therefore, this will contain the 1st and 2nd digits of the address. |
|  | R772 | Sets the lower byte of the station address assigned to the bottom communication port. This only contains one digit, which is the 3rd digit of the address. |
|  | R773 | Sets the baud rate for the bottom communication port. |
|  | R774 | Sets the leading communications delay time for the bottom communication port. |
|  | R775 | Sets the trailing communications delay time for the bottom communication port. |
|  | R776 | Sets the leading communications delay time for the top communication port. |
|  | R777 | Sets the trailing communications delay time for the top communication port. |

## Programming Basics

In This Chapter. . . .

- Introduction
- Using Boolean Instructions
- Using Timers
- Using Counters
- Using the Accumulator respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.


## Introduction

This chapter describes some basic programming concepts used with the DL305 CPUs. It doesn't provide detailed information on each instruction, but instead shows how you can use the most basic elements of the instruction set. If you have quite a bit of PLC programming experience, you may already know some of the information. However, we suggest you at least read the portion that discusses the accumulator operation. The accumulator is used in many different operations.
This chapter provides an overview of the following programming concepts.

1. Boolean Instructions
2. Timer Instructions
3. Counter Instructions
4. Shift Register Instruction
5. Accumulator Instructions

Detailed examples of all categories of instructions are included in Chapters 11 \& 12. The DL305 CPUs can be programmed with the DirectSOFT PC-based programming package, or by using the DL305 handheld programmer. There is a separate manual available for each of these products. If your are not familiar with the chosen programming device we recommend you use the appropriate programming device manual along with this manual to program your DL305 system.
The following examples will help you understand how DL305 instructions are put together to create a program solution.

## Using Boolean Instructions

Do you ever wonder why so many PLC manufacturers always quote the scan time for a 1 K boolean program? Simple. Most all programs utilize many boolean instructions. These are typically very simple instructions designed to join input and output contacts in various series and parallel combinations. Since the DirectSOFT package allows you to use graphic symbols to build the program, you don't absolutely have to know the boolean equivalents of the instructions. However, it may be helpful at some point, especially if you ever have to troubleshoot the program with a Handheld Programmer.
The following paragraphs show how these boolean instructions are used to build simple ladder programs.

## END Statement

## Simple Rungs

## Normally Closed

 ContactAll DL305 programs require an END statement as the last instruction. This tells the CPU this is the end of the program. Any instructions placed after the END statement will not be executed. (This can be useful in some cases. See Chapter 13 for an example.)


You use a contact to start rungs that contain both contacts and coils. The boolean instruction that does this is called a Store or, STR instruction. The output point is represented by the Output or, OUT instruction. The following example shows how to enter a single contact and a single output coil.


Normally closed contacts are also very common. This is accomplished with the Store Not or, STRN instruction. The following example shows a simple rung with a normally closed contact.

DirectSOFT Example Handheld Mnemonics


Contacts in Series Use the AND instruction to join two or more contacts in series. The following example shows two contacts in series and a single output coil.


Handheld Mnemonics

STR 000
AND 001
OUT 020 END

Midline Outputs Sometimes it is necessary to use midline outputs to get additional outputs that are conditional on other contacts. The following example shows how you can use the AND instruction to continue a rung with more conditional outputs.


Parallel Elements You may also join contacts in parallel. The OR instruction allows you to do this. The following example shows two contacts in parallel and a single output coil.


## Joining Series Branches in Parallel

Quite often it is necessary to join several groups of series elements in parallel. The Or Store (ORSTR) instruction allows this operation. The following example shows a simple network consisting of series elements joined in parallel.


Joining Parallel Branches in Series

Quite often it is also necessary to join one or more parallel branches in series. The And Store (ANDSTR) instruction allows this operation. The following example shows a simple network with contact branches in series with parallel contacts.


## Comparative Boolean

Many applications require comparisons of data values. This is especially true in applications that use counters. Some PLC manufacturers make it really difficult to do a simple comparison of a counter value and a constant or register. The DL330 and DL340 CPUs provide Comparative Boolean instructions that allow you to quickly and easily solve this problem. Comparative Boolean evaluates two 4-digit values using boolean contacts. The valid evaluations are equal and not equal.
In the following example when the value in counter C600 is equal to the constant value 1234 , output 020 will energize.


The DL330P also provides Comparative Boolean instructions, but they are greater than and less than instructions instead of equal and not equal.

## Combination Networks

Boolean Stack

You can combine the various types of series and parallel branches to solve most any application problem. The following example shows a simple combination network.


There are limits to how many elements you can include in a rung. This is because the DL305 CPUs use an 8 -level boolean stack to evaluate the various logic elements. The boolean stack is a temporary storage area that solves the logic for the rung. Each time you enter a STR instruction, the instruction is placed on the top of the boolean stack. Any other instructions on the boolean stack are pushed down a level. The AND, OR, ANDSTR, and ORSTR instructions combine levels of the boolean stack when they are encountered. Since the boolean stack is only eight levels, an error will occur if the CPU encounters a rung that uses more than the eight levels of the boolean stack.

All of you software programmers may be saying, "I use DirectSOFT, so I don't need to know how the stack works." Not quite true. Even though you can build the network with the graphic symbols, the limits of the CPU are still the same. If the stack limit is exceeded when the program is compiled, an error will occur.

The following example shows how the boolean stack is used to solve boolean logic.


ANDSTR

| 1 | 000 AND (NOT 005 OR 004) AND [001 OR (002 AND 003)] |
| :--- | :--- |
| 2 |  |
| 3 |  |

8

Programming Basics

## Using Timers

Timers are used to time an event for a desired length of time. The single input timer will time as long as the input is on. When the input changes from on to off the timer current value is reset to 0 . Timers normally time in tenth of a second intervals, but you can turn on Special Relay 770 to change the timers to hundredth of a second intervals. There is discrete bit associated with each timer to indicate the current value is equal to or greater than the preset value. The timing diagram below shows the relationship between the timer input, associated discrete bit, current value, and timer preset.



## Using Counters

Counters are used to count events. There are two types of counters.

- Regular Up counters
- Stage counters (used with the RLLPLUS instructions)

The up counter has two inputs, a count input and a reset input. The maximum count value is 9999 . The timing diagram below shows the relationship between the counter input, counter reset, associated discrete bit, current value, and counter preset.


The stage counter has a count input and is reset by the RST instruction. This instruction is used with the RLLPLUS instructions. The maximum count value is 9999. The timing diagram below shows the relationship between the counter input, associated discrete bit, current value, counter preset and reset instruction.


Counter preset

## Using the Accumulator

The accumulator in the DL305 series CPUs is a 16 bit register which is used as a temporary storage location for data being copied or manipulated in some manor. For example, you have to use the accumulator to perform math operations such as add, subtract, multiply, etc. Since there are 16 bits, you can use up to a 4 -digit BCD number. The accumulator is reset to 0 at the end of every CPU scan.
Copying Data to and from the Accumulator

The Data Store (DSTR) and Data Out (DOUT) instructions and their variations are used to copy data from a register location to the accumulator, or to copy data from the accumulator to a register location.
In the following example, when input 000 is on the value (7502) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The value in the accumulator is output to data registers R404 and R405 using the Data Out (F60) instruction.


You probably noticed it took two registers to hold a 4-digit BCD number. This is because each BCD digit requires four binary bit positions.

Since the accumulator is 16 bits and register locations are 8 bits, there are variations of the DSTR and DOUT instructions that allow you to copy a single register, or even half of a register ( 4 bits) either to or from the accumulator. The following example shows how you could use the DSTR3 and DOUT2 instructions to copy the lower 4 bits from register 5 to the upper 4 bits of register 16. (These registers correspond to I/O points and Control Relays respectively.)

## DirectSOFT Display



Changing the Accumulator Data

Instructions that change or manipulate data in some way also use the accumulator. The result of the change resides in the accumulator. The original data that was being changed is cleared from the accumulator. In the following example, when input 000 is on the value in R000 and R010 is loaded into the accumulator using the Data Store 5 (F55) instruction. The bit pattern in the accumulator is shifted to the left 4 bit positions using the Shift Left (F80) instruction. Notice how the result resides in the accumulator. The value in the accumulator is copied to data registers R404 and R405 using the Data Out (F60) instruction.


Programming Basics

## Accumulator Operations

The following table lists several instructions that utilize the accumulator. Not all instructions allow you to use all the different memory types. Chapters 11 \& 12 provide details on these instructions.

| Category | Mnemonic | Description | Memory Areas |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | I/O | CRs | Data Register | Current Values | $\begin{aligned} & \text { 4-digit } \\ & \text { BCD } \\ & \text { Const. } \end{aligned}$ | Shift Register Coils |
| Data Load | $\begin{aligned} & \hline \text { DSTR } \\ & \text { (F50) } \end{aligned}$ | Load a 4-digit constant or a 2-bytes of register data into the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DSTR } 1 \\ & \text { (F51) } \end{aligned}$ | Load 1-byte of register data into the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DSTR } 2 \\ & \text { (F52) } \end{aligned}$ | Load the upper 4 bits of a register into the lower 4 bits of the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DSTR } 3 \\ & \text { (F53) } \end{aligned}$ | Load the lower 4 bits of a register into the upper 4 bits of the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
|  | $\begin{aligned} & \hline \text { DSTR } 5 \\ & \text { (F55) } \end{aligned}$ | Load the digital values of 16 I/O points (2 bytes) into the accumulator | $\bigcirc$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
| Data Out | $\begin{aligned} & \text { DOUT } \\ & \text { (F60) } \end{aligned}$ | Write the accumulator to 2 sequential registers | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DOUT } 1 \\ & \text { (F61) } \end{aligned}$ | Write the lower byte of the accumulator to a register | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DOUT } 2 \\ & \text { (F62) } \end{aligned}$ | Write the lower 4 bits of the accumulator to the upper 4 bits of a register | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DOUT } 3 \\ & \text { (F63) } \end{aligned}$ | Write the lower 4 bits of the accumulator to the lower 4 bits of a register | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DOUT } 5 \\ & \text { (F65) } \end{aligned}$ | Write the contents of the accumulator to a 16-point output module (2 bytes) | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\times$ | $\times$ | $\bigcirc$ |
| Math | $\begin{aligned} & \hline \text { CMP } \\ & \text { (F70) } \end{aligned}$ | Compare a 2-byte BCD reference or a 4-digit BCD constant to the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{aligned} & \hline \text { ADD } \\ & \text { (F71) } \end{aligned}$ | Add a 2-byte BCD reference or a 4-digit BCD constant to the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{array}{\|l\|} \hline \begin{array}{l} \text { SUBTRACT } \\ \text { (F72) } \end{array} \\ \hline \end{array}$ | Subtract a 2-byte BCD reference or a 4-digit BCD constant from the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{array}{\|l} \hline \begin{array}{l} \text { MULTIPLY } \\ \text { (F73) } \end{array} \\ \hline \end{array}$ | Multiply a 2-byte BCD reference or a 4-digit BCD constant by the value in the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{aligned} & \text { DIVIDE } \\ & \text { (F74) } \end{aligned}$ | Divide the accumulator by a 2-byte BCD reference or a 4-digit BCD constant | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |

O - Memory Type available for use with the instruction
X — Not available

| Category | Mnemonic | Description | Memory Areas |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | 1/0 | CRs | Data Register | Current Values | $\begin{aligned} & \text { 4-digit } \\ & \text { BCD } \end{aligned}$ Const. | Shift Register Coils Coils |
| Bit <br> Manipulation | $\begin{aligned} & \hline \text { DAND } \\ & \text { (F75) } \end{aligned}$ | Performs a bit "AND" on a 2-byte reference or a 4-digit BCD constant and the bits in the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{array}{\|l\|} \hline \text { DOR } \\ \text { (F76) } \end{array}$ | Performs a bit "OR" on a 2-byte reference or a 4-digit BCD constant and the bits in the accumulator | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ | $\bigcirc$ |
|  | $\begin{array}{\|l} \hline \text { SHIFT } \\ \text { RIGHT } \\ \text { (F80) } \end{array}$ | Shifts the contents of the accumulator to the right a specified number of times. 1-15 bits can be shifted. | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
|  | $\begin{aligned} & \hline \text { SHIFT LEFT } \\ & \text { (F81) } \end{aligned}$ | Shifts the contents of the accumulator to the left a specified number of times. 1-15 bits can be shifted. | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
| Data <br> Conversion | $\begin{aligned} & \hline \hline \text { DECODE } \\ & \text { (F82) } \end{aligned}$ | Decodes the first 4 bits of the accumulator into a decimal number. | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
|  | $\begin{array}{\|l} \hline \text { ENCODE } \\ \text { (F83) } \end{array}$ | Encodes an accumulator bit into a 4-bit code that represents the decimal number ( $0-15$ ). | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
|  | $\begin{array}{\|l\|} \hline \text { INV } \\ (\text { F84 }) \end{array}$ | Logically inverts the contents of the accumulator ( 1 to 0,0 to 1 ). | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
|  | $\begin{aligned} & \text { BCD-BIN } \\ & \text { (F85) } \end{aligned}$ | Converts the accumulator value from BCD to Binary | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
|  | $\begin{aligned} & \begin{array}{l} \text { BIN-BCD } \\ \text { (F86) } \end{array} \end{aligned}$ | Converts the accumulator value from Binary to BCD | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |
| Fault Detection | $\begin{array}{\|l} \hline \text { FAULT } \\ \text { (F20) } \end{array}$ | Sends a 4-digit BCD number, from a 2-byte reference or a constant, to the programmer display | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ |

o - Memory Type available for use with the instruction
X - Not available

## RLLPLUS

## Programming Basics

In This Chapter. . . .

- Introduction
- An Example Machine
- An RLL Solution
- An RLLPLUS Solution
- Stage Instruction Execution
- Activating Stages
- Using Outputs in Stages
- Using Timers and Counters in Stages
- Using Data Instructions in Stages
- Using Comparative Contacts in Stages
- Parallel Branching Concepts
- Unusual Operations in Stages
- Two Ways to View RLLPLUS Programs
- Designing a Program Using RLL PLUS Instructions


## Introduction

If you've ever been around some really accomplished RLL programmers you have probably been amazed at how easily they seem to be able to create programs of incredible complexity. Well, not everyone has years of experience in programming PLCs. Because of this the DL330P CPU has RLLPLUS instructions that make it considerably easier to design and create programming solutions. These instructions are especially useful to those of you who aren't that familiar with the interlocking concepts commonly used in RLL programs.
You can still use the normal instructions you've already seen, plus you only have to become familiar with a few new instructions that help you organize your program into manageable pieces.
This programming method is similar to Sequential Function Chart programming and literally allows you to design a flowchart of the program operation sequence and load it into the CPU! You can expect to see several benefits by using this method.

- Considerably reduced program design time. We've seen many, many cases where these few instructions have cut program design time by well over 50\%.
- Shorter, more simple programs. Later in this chapter we'll show you why your programs sometimes end up being a lot larger than you first anticipated. The RLLPLUS instructions can help make your programs simple for everyone to understand.
- Easier program troubleshooting. How many times have you tried to troubleshoot or modify a program that was written by someone else? If you've done this very often you know it's not an easy task. This chapter will show you a few instructions that will also help with this problem as well.

The following paragraphs discuss several RLLPLUS programming concepts. We'll use a simple example to show you how to use the various types of instructions. Also, we'll show you the equivalent program without RLL PLUS instructions to give you an idea of the differences between the two approaches.

NOTE: The DL330P has several instructions that do not operate quite the same as the equivalent instructions in the DL330 or DL340. If you want to take advantage of the benefits associated with the RLLPLUS instructions, make sure you also take time to review Chapter 12. This chapter discusses the instructions that are unique or different with the DL330P CPU.

## An Example Machine

Machine Operation Most any application can be described as a sequence of events. The PLC program merely makes sure the events are completed in a specific order. Not only does the program control normal operation, but it also has to allow for machine failures and emergency conditions. Consider a simple example.

1. The operator presses the start switch.
2. The machine checks for a part. If the part is present, the process continues. If not, the conveyor moves until a part is present.
3. The part is locked in place with a clamp.
4. The press stamps the part.
5. The clamp is unlocked and the
 finished piece is moved out of the press.
6. The process stops if the machine is in one-cycle mode, or the process continues if automatic mode is selected.

Machine Flowchart The following diagram provides a flowchart of this operations sequence.


## An RLL Solution

Why is RLL so popular? Simple. Before the PLC arrived control problems were generally solved with hardwired relays and switches. About 30 years ago people started experimenting with a way to make quick and easy changes without changing the actual panel wiring. Thus, the PLC was born. Since the people developing and using this new technology were familiar with the relay and switch solution, it made sense to have this new technology emulate something that was familiar to them. That's why RLL programs emulate a relay panel solution.
When you supply power to a relay panel the combination of contact and coil status determines what actions take place. Since the RLL program emulates the relay panel solution, the entire program is scanned left-to-right, top-to-bottom. The program executes the operations sequence when a certain combination of contacts are activated. This process is known as interlocking.
Since many PLCs do not have instructions to help manage the operations sequence, the programmer has to make sure the program carries out the correct sequence by adding the required interlocks. One great thing about the RLL solution is the individual rungs are easy to understand. By examining the contacts you can easily determine if the output will be on or off.


Many accomplished RLL programmers use things such as Master Control Relays and Subroutines to reduce the amount of interlocking required. However, these instructions can sometimes make the program more difficult to understand. There are several things you should notice about our simple press program.

- Most all rungs use some amount of interlocking.
- The number of interlocks is usually proportional to the number of tasks in the operations sequence.
- Most of the instructions are devoted to processing the interlocks. (Plus, since the program is larger, it takes more time to process.)
- It usually requires several attempts until a program is designed that is not susceptible to inadvertent activation and deactivation.
- The program can be difficult to debug if you do not have a considerable amount of RLL programming experience.


## An RLL PLUS Solution

The RLLPLUS instructions keep the simplicity of the contacts and coils while removing some of the problems associated with the enormous amount of interlocks. There are several RLLPLUS instructions, but the most often used are the Initial Stage (ISG), Stage (SG), and Jump (JMP) instructions. Here's the example press program created using RLLPLUS instructions. There are two things you should notice.

- Control Relay interlocks are not required.
- The program directly follows the flowchart of the press operation.
How can this happen? Simple. The interlocks were added to the RLL program to keep the outputs from coming on at the improper time. This is because every rung of the RLL program was examined on every scan.
The Stage instructions (and the logic between the Stage instruction and the next stage instruction) are not necessarily examined on every scan. Only stages that are on are examined.
Each stage instruction has a status bit that is on when the stage is active, and off when the stage is inactive. On every scan the CPU examines which stage status bits are on and only examines the logic in those stages. If a stage is inactive, the CPU skips the logic between that stage and the next active stage.
The following pages will talk about several different aspects of the CPU execution for the Stage Instructions. It will help to understand the pieces of an individual stage.


## Stage Nomenclature

As we discuss the examples it will be necessary for you to understand the various pieces that can make up a program stage.

- Stages - a instruction that denotes a piece of the program
- Actions- an event in the program, such as an output, jump, or some other instruction.
- Transitions - the event that causes the program to move to the next stage.


## Stage Instruction Execution

Stage Instruction Numbering

Stages are numbered in octal, so you can't have any stages with the numbers 8
or 9 in them. Notice the stages skipped from 7 to 10 since the numbers 8 and 9 are not used. There are 128 (decimal) stages available in the DL330P CPU, numbered 0 through 177.
Since each stage has a unique status bit, you cannot have stages with the same address number. For example, since the example program already has a Stage1, we wouldn't want to use that number again.
There's another advantage to having a status bit for each stage. This allows you to skip stage numbers as necessary. This is a good practice to follow because it makes it easier to insert stages later without affecting the appearance of the program flow.
The stage numbers do not necessarily have to be numbered sequentially, but it can be extremely helpful to use sequential numbers if you are working with large programs.
Also, the stages do not have to be entered sequentially with the programming device. For example, you could have Stage 100 be the first entry in the program. This is not a good programming practice, but since the CPU looks at the active status bits to determine which stages to execute, it doesn't care where the stages are physically located in the program.


NOTE: Remember, machines do break. We recommend you use numbering that matches the machine flowchart. Also, we recommend you enter the program in the same order whenever possible. This will make troubleshooting much easier.

The section on Designing an RLLPLUS Program at the end of this chapter provides guidelines for assigning numbers to the stage instructions.

## Octal Numbering



Skip Numbers if Necessary


Non-sequential Numbering


A Few
Simple Rules for
Execution

Since the CPU will only examine the logic in those stages that are active, it is important you understand how stages can be turned on and off. There are a few simple rules that dictate how this works. This may seem like quite a few things to remember, but it's really pretty simple. We'll show examples in the following pages that show how each of these rules apply to the program execution.

1. Only active stages are executed. If a stage is inactive, the CPU skips the logic between that stage and the next active stage.
2. You can turn stages on by the following methods.
2.a Initial Stages are automatically turned on when the CPU transitions from Program Mode to Run Mode.
2.b A stage can be turned on when the program "jumps" from stage to stage with the Jump (JMP) instruction.
2.c You can use the SET instruction to set a stage status bit just like you would SET an output.
2.d A stage can be turned on when the program has power flow between two stages that are tied together by a single transition element.
3. You can turn stages off by the following methods.
3.a An active stage is automatically turned off if the program jumps from the active stage to another stage.
3.b You can use the Reset (RST) instruction to turn off a stage just like you use Reset to turn off an output point.
3.c The current stage is automatically turned off if the program has power flow between the current stage and the next stage.

## Activating Stages

Using Initial Stages Any initial stages (ISG instructions) are automatically turned on when the CPU goes from Program Mode to Run Mode. For example, when the CPU executing our example program enters Run Mode, the Initial Stage (ISG 000) will be turned on automatically. The other stages are off, so the CPU only scans the portion of the program associated with ISG 000.
Since there's only one rung in Stage 0, the CPU continually monitors the start switch. Nothing else will happen until the
 start switch is pressed.

Although it is unusual, there may be times when you need more than one initial stage. There is nothing at all wrong with this. If your application has a need for more than one starting point, you can use more than one initial stage. For example, if you had three initial stages, then those three stages would all be active when the CPU entered the Run Mode.

## Using Jump Instructions

When the operator presses the start switch input 000 comes on. When 000 comes on the CPU executes the Jump instruction and "jumps" to Stage 1.
Now the CPU only scans Stage 1. Stage 0 is no longer scanned after the program jumped to stage 1. This means the Jump instruction did two things.

- It activated the destination stage. In this case, it activated stage 1.
- It deactivated the stage it came from, which was stage 0 in this case.
So, you can jump to a stage to turn it on, and when you jump from a stage it turns off.


This example only shows an action that initiates a jump to one destination. You can use several jumps ORed together if necessary. Examples of this will be shown later.
There's also another type of Jump instruction called a Not Jump. This instruction only works if the input conditions are not true, whereas the regular JMP instruction only works if the input conditions are true.
In the previous example we examined a single contact to determine which part of the program to jump to next. If the part is present (001 closed), the program jumps to Stage 2. If a part is not present (001 open), the program jumped to Stage 5. We could have used a single contact and the NJMP instruction.
The program example to the right shows how the NJMP instruction would be used in this situation. Notice there is one less instruction required in this example compared to the previous one.

NOTE: We strongly recommend you avoid using the NJMP instruction. This is because program debugging can become more difficult, especially for those who are not so familiar with structured programming concepts.

## Using Set Instructions with Stages

When you examine the instruction set more carefully you'll notice the DL330P CPU offers a Set (SET) instruction that works similarly to a latching operation. For example, you could use a SET instruction to latch an output point. The output point can then be unlatched with the Reset (RST) instruction.
You can also use a SET instruction to turn on a stage. To show how this works, we're going to add a stage to the program. You may have noticed the original flowchart did not contain a stop switch. Well, we don't want to make these little widgets forever, so we're adding Stage 150, which monitors for a stop switch. (This is also a good example of how you can skip stage numbers.)
Notice we added a SET instruction in the first stage. Now when the start switch is pressed, two stages will be activated. The CPU examines Stage 1, which monitors for a part, and it also examines Stage 150, which monitors the stop switch.


We did not absolutely have to use a SET instruction in the example. We could have used a Jump, since you can jump to more than one stage. We just used a SET to show how it works.
If you examine Stage 150, you'll notice we do three things when the stop switch is pressed.

- The RST 020-022 instruction makes sure all the outputs are turned off. (We'll discuss this in more detail in the next section.)
- The RST S0-S6 instruction resets (turns off) stages 0 through 6. We reset the entire range so that we guarantee we can stop the press no matter which stage is currently executing. Notice we reset stages that were not necessarily turned on with the SET instruction. The Reset (RST) instruction can be used to turn off stages, no matter how they were turned on. This is especially handy in larger, more complex programs.
- The program jumps back to Stage 0 and starts over again. Note, just because Stage 0 is an initial stage does not mean it can only be active at a transition to Run Mode. You can return to an Initial Stage at any time. It's just the CPU automatically activates Initial Stages at the Run Mode transition.


## Power Flow Transitions

You do not always have to use a Jump instruction to move from stage to stage. If you only move to one stage, instead of multiple stages, you can use what it is called a power flow transition. For example, we used Jump instructions in our sample program. For those stages that did not have multiple transition possibilities, we could have just used power flow transitions.
Look at Stage 2. Notice how the transition contact, 002 now is directly connected to the next stage, Stage 3. You can only do this if you are moving from one stage to one other stage.
If you examine Stage 1, you'll notice we have to use the Jump instructions because the program can transition to more than one stage.


NOTE: We suggest you use Jump Instructions instead of power flow transitions. This is because we've seen many cases where we had to come back and add things to the program. If you used Jumps from the beginning, you only have to add another Jump instruction. If you used power flow transitions, the program edits can take a little longer.

## Using Outputs in Stages

## Setting Outputs with the SET Instruction

Since the CPU only examines the logic in stages that are on, you have a lot more flexibility in how you use outputs with the RLLPLUS instructions. Also, you don't have to worry about adding several permissive contacts to keep the output from coming on at an inappropriate time. (If the stage is not on,the CPU doesn't even scan the stage, so the output can't possibly be turned on by the logic in that stage.)
If you examine Stage 2, you'll notice we use a SET instruction to clamp the part in place. Why a set? Simple. If we used a regular output the clamp will be deactivated when the program transitions to Stage 3. Remember, when you leave a stage the CPU no longer scans that stage until it is turned on again. So if we had used a regular OUT instruction, the CPU would have automatically turned off the output, which would have unclamped the part.
The first example shows the program execution in Stage 2. The second example shows what happens on the next scan after the part is locked. Notice the clamp output is still on even though the CPU is not scanning this portion of the program. This is why we use the SET instruction in this case. We want the clamp to stay on while the press completes the cycle.
The clamp will stay on until the program enters Stage 4. Stage 4 unlocks the part by resetting output 020 when the press returns to the top limit.


## Using the OUT Instruction

One other benefit with RLLPLUS is the ability to use the same output in multiple places. Instead of using the SET instruction in Stage 2, we could have just put the clamp output, 020, in all the stages where we wanted the part to remain clamped.
If you examine Stage 2 you'll notice output 020 is on because the stage is active. The next example shows what happens after the part is locked in place. The program moves to Stage 3 from Stage 2. Notice output 020 is now off in Stage 2. However, since we included the same clamp output in Stage 3, the part remains clamped in place.
The clamp will automatically turn off when the program enters Stage 4. Notice Stage 4 does not have to have any kind of Reset instruction, since the output is automatically turned off when the program exits Stage 3.
The concept of automatically turning off the outputs sometimes confuses many people. However, the CPU just uses a very simple algorithm to determine if the output should be turned off. The following diagram shows how this algorithm works.



## Latching Outputs with Stages

There's one more way to control outputs with the Stage instructions. You may recall once a stage is turned on, you can only turn it off by resetting it, or by having a transition from it, either by a Jump or a power flow.
What happens if you have a stage that does not have any kind of transition? What if it doesn't have a Jump instruction or any other kind of transition contact leading to another stage? Simple. The stage will stay on until it is reset by some other part of the program that uses a Reset instruction.
This makes it easy to use a stage without a transition to latch an output. For example, if you examine Stage 2 you'll notice we've now changed this part of the program again. Now this stage sets Stage 140 ,which will be used to control the clamp.
Notice Stage 140 does not have any type of transition. The only way to turn off the clamp is to Reset Stage 140. This instruction has now been included in Stage 4. So, after the program transitions to Stage 4, the Reset instruction will turn off Stage 140.


## Using Timers and Counters in Stages

Time Based
Transitions

Up to this point we've been using certain events that triggered the transition from stage to stage. There will probably be many cases where the transition should be related to a timer value. For example, if you know the speed of the conveyor you could use a timer to control the conveyor movement.
If we used this approach we would modify Stage 5 as shown. Notice the timer does not have a preset value. The timer begins incrementing as soon as it becomes active. Since the timer does not have a preset value, you do not have a timer contact, so you have to use a comparative instruction.
In the example shown, the conveyor will be turned on for 5 seconds and then the program will jump to the next stage.
$\xrightarrow{\text { Only executes logic in stages that are active }}$


Using Counters

There will also be times when you need to count things that happen throughout the process. For example, you may want to know the number of parts produced during any given shift, or you may know the presses generally require some type of maintenance after a certain number of cycles.
If we wanted to count the number of widgets made on our simple press, we could just add a counter to Stage 4 to monitor how many times the press is used. We're also going to use the counter as an automatic shutdown when the press has made 5000 parts so we've added a new rung in Stage 150 to perform the shutdown operation.
Notice the counter does not have a reset leg. This is true only when you use a counter with the DL330P. (The other CPUs have counters with reset legs.) Even though this counter does not have a reset leg, it can be reset with a Reset instruction. This works just like an output reset, so you could place this reset wherever it is appropriate. We've placed it in Stage 150 for this example.
When the parts count reaches 5000, the program will finish the current cycle, reset the part counter, and jump to Stage 0 to wait for another start cycle. You may notice we added an additional input, 006. This is what allows the program to finish the current cycle. (You may recall 006 only came on after the part was unlocked and the conveyor was indexed.)


## Using Data Instructions in Stages

Even though there are a few differences in the way some of the instructions operate between the various CPUs, there are many of the normal instructions that can be used inside an individual stage. For example, you may need to load data into the accumulator to perform some type of math, or, you may need to store values into register locations.
If you examine Stage 3, you'll notice we've added a couple of instructions. These instructions store the current parts count in a register.
Now the CPU will take the current parts count, stored in R600, and load it into the accumulator with the DSTR instruction. Then this 4-digit BCD count will be moved to R400 with the DOUT instruction.
This is just one example of how you can use the various types of data instructions. There are many other possibilities. Just remember, if the stage is active, the instructions can be executed. If the stage isn't active, the instructions will not even be examined.


## Using Comparative Contacts in Stages

You may recall you had to use a comparative instruction with the timers and counters. The DL330P provides several comparative contacts that are very useful. You can use these contacts to examine the relationship between a counter or timer value and a constant or register value.
For example, let's assume the pressed widgets move off the conveyor into a holding bin. The bin can only hold 1000 widgets, so we'll add another counter, C601, to note how many widgets are in the bin. Also, we want to use different colored lights mounted on top of the press to alert a forklift driver the bin needs to be carried to the next operation. We'll use the following indicators.

| Indicator | Meaning | Address | Stage |
| :--- | :--- | :--- | :--- |
| Green | OK | 040 | 21 |
| Yellow | Soon | 041 | 22 |
| Red | Urgent | 042 | 23 |
| Reset | Emptied | 030 |  |

Notice we've added a few more stages to monitor this condition. For this example, assume the press has made 750 widgets. This means the Yellow indicator (Stage 22) should be active.
We also need a way to reset the bin counter whenever the forklift driver empties the bin. If you examine Stage 21 through Stage 23, you'll notice we reset the bin counter whenever the bin reset (030) is active.

This example doesn't show it, but you would also have to make some changes to other parts of the program. For example, you'd need to modify the Stop Stage to shut off these stages when the machine was stopped.


## Parallel Branching Concepts

Branching Methods

As you examined some of the previous examples you saw we could have more than one stage being processed on any given scan. The CPU scanned the first active stage and then moved on to the next active stage, skipping any inactive stages in between. For some complex applications, you can easily have as many parallel paths as necessary. This is often called branching or divergence.
There are a couple of approaches you can take when you want to turn on more than one stage. The diagrams shown don't necessarily apply to our press example, but instead show the various approaches.
In this example, you use one transition contact to activate several stages.

- The SET instruction sets a range of stages. These stages would remain on until they were reset, or, until any transition instructions contained within the stages were executed.
- There are two Jump instructions, both activating different stages.

In this example, notice the stage that gets activated depends on an extra condition. For example, if the machine was capable of producing three different patterns, there may be a section of program for each pattern.
There are other types of contacts that can be used. For example, you may recall we used Comparative contacts in some earlier examples.
Notice we had to repeat the start switch in a separate rung each time. At first glance you would think you could simply have one Start switch contact and OR the remaining switches. The DL305 CPUs do support midline outputs (which is what this is called), but only in an AND situation.



Joining Parallel There are many times you have to bring parallel branches back together at some Branches point in the program. You may recall the stages have status bits associated with them. You can use this status bit as a contact to easily converge the parallel paths.
To illustrate this method, we're going to use a simple press with two stations. Now a widget must get pressed at each station before it is a finished product. Since there are two stations, we must make sure both operations are complete before we move the conveyor.

Here's a flowchart that describes the two-station press. Please note we've changed some of the stage numbers, input numbers, and output numbers, so they won't necessarily match the previous examples.


You've already seen how the basic sequence of operations was executed. so we're only going to show the portions of the program that describe how the branches are joined together.

## Using Stage Bits as Contacts

If you examine the flowchart you'll notice once the part is unclamped in station B, the program transitions to Stage 33 which indicates Station B is complete.
If you look at that portion of the program shown here, you'll notice there are no other instructions or actions that take place in this stage. This is why we call it a "dummy" stage. We're just going to use the status of the stage bit associated with this dummy stage as a contact elsewhere in the program to indicate station $B$ is finished.
You may be wondering how we can turn off this stage. Since it does not have any type of jump or power flow transition, the only other option is to Reset the stage. We'll do this later in the program.


Stage Contact
Example

Since each stage has a status bit that is either on or off, you can use this bit as a contact in the program. If you examine Stage 23 you'll notice we've used a contact labeled S33. This contact reflects the status of Stage 33, which indicated Station B was finished.
When S33 is on, the contact labeled S33 is also on and the program will transition to Stage 40. In Stage 40 we use a reset instruction to reset Stage 33 before we move the conveyor.


## Unusual Operations in Stages

Using the Same Output Multiple Times

Over the last few pages you've learned how the CPU executes the Stage instructions. However, there are a few unusual circumstances that may not work exactly as the appear.
In the program shown it appears output 021 will be turned on at three separate times before the program jumps to the next stage. However, the only time the output actually comes on is when the final condition has been met.
Why? Remember if you use multiple outputs in a program, the last rung containing the output controls the status that will be written to the module. This is no different in a program that uses RLLPLUS instructions.
In this example, the last comparison rung says the output should be off until the
 timer value reaches 90 seconds.

In the previous example the same output was used multiple times in the same stage. The last use of the output controlled the status of the output.
There may be occasions when you have the same output in different stages. Even though it's not advisable to do this in normal RLL programs, this is perfectly acceptable with a program that uses RLLPLUS instructions. However, if both stages are active at the same time, then the logic in the last stage will control the status of the output.
In the example shown, if both stages are active, then the logic in Stage 70 will control the output status.


Using a Set Out Reset (SET OUT RST) Instruction

Many normal RLL programs use one-shot instructions. In the DL305 instruction set, this instruction is called a Set Out Reset (SET OUT RST).
In the program shown, input 001 will trigger the SET OUT RST 160 instruction, which will in turn activate output 021 for one scan.
However, what happens if 001 stays on and Stage 10 is activated, deactivated, and then activated again? At first glance it appears the one shot only gets executed one time since 001 stayed on while Stage 10 was turning on and off. It doesn't work this way.
The logic in an inactive stage is not executed. So even though the stage became active the SET OUT RST instruction did not see an off to on transition, so the instruction is not executed.
The SET OUT RST instruction will work in an active stage as long as the input transitions from off to on while the stage is active.

Output Placement As you've seen in some of the previous examples, we always place unconditional outputs immediately following the Stage Instructions. There's a reason for doing this.
If you look at the example stage shown here, the output is placed after a counter box. The DirectSOFT software and the Handheld Programmer will allow you to enter this as shown. However, the CPU will only turn on output 021 when the counter input 001 is turned on. This is because the CPU interprets the output as being tied to the counter input leg instead of the Stage power rail.
You can easily avoid this problem by placing any unconditional actions at the very beginning of the stage. Then, the output will work the way you expect.


Incorrect Placement


## Correct Placement



RLL PLUS Programming Basics

## Two Ways to View RLL PLUS Programs

Throughout the example programs, we've consistently shown how the instructions appear in when viewed as ladder instructions. However, with DirectSOFT, you also have the capability to view the program as a flowchart. You can even view the program flowchart (in Stage View) and view the ladder program at the same time with a split screen feature. The DirectSOFT manual provides detailed information on how to view the programs in this manner.
DirectSOFT
Stage View


## DirectSOFT

Ladder View


## Designing a Program Using RLLPLUS Instructions

As with most any application problem, a thorough understanding of the tasks combined with a good plan of execution often results in success. The RLLPLUS instructions provide an easy way to load the plan of execution directly into the CPU. The easiest way to make sure you understand the tasks is to make a flowchart. This is often the most critical part of creating a program that uses RLLPLUS instructions. There are a few simple steps you can follow to create a detailed flowchart.

1. Create a top-level flowchart.
2. Expand the flowchart by adding things that cause the transitions from step to step.
3. Add any actions that must occur in each step.
4. Add any conditions that control the actions.
5. Add any special monitoring or alarm steps that must be performed.
6. Assign numbers to the stages (steps).
7. Add the I/O instructions and addresses (input contacts, output coils, jump instructions, etc.)
8. Enter the program.

Use DirectSOFT to Save Time

The DirectSOFT programming package allows you to quickly and easily create programs with RLLPLUS instructions. The software has special features that allow you to create the flowcharts, add the transitions, actions, etc. Even if your programs are fairly small, DirectSOFT can make the job much easier.

Step 1:
Design a Top-level Flowchart

There are many different ways to design a flowchart of the application problem, but there are a few guidelines that will make the job easier.

1. Start with a "top-level" flowchart that breaks the operation sequence into simple pieces.
2. Each piece of the top-level flowchart should only represent one action. Resist the temptation to group several operations into one part of the flowchart.
3. Don't try to add input or output addresses to the flowchart. Only use words to describe the things that are taking place.
4. Don't worry about special conditions, such as stop conditions, alarms, etc at this point. These will be added later when you fully understand how the main part of the operations sequence is organized.
You can draw the flowchart horizontally or vertically at any point in the design process, the choice is yours. Here's an example top level flowchart for our simple one-station press.


Step 2:
Add Flowchart Transitions

Once you have designed the basic operating sequence you should determine the events that cause a transition from step to step. During this phase you may find things need to be added to the flowchart. All you're really doing is adding more detail to the top-level flowchart. Once again, don't try to use addresses yet. Concentrate on using words to describe the events taking place. The following flowchart adds the transition conditions for our one-station press.


RLLPLUS Programming Basics

Step 3:
Add Actions

After you determine the events that cause a transition from step to step you should add any actions that need to take place during the sequence. Again, don't try to use addresses yet. Concentrate on using words to describe the actions taking place. The following flowchart adds the actions that take place during each part of the program.


Step 4:
Add Conditions for Actions

Some actions may only take place if certain conditions are met. Examine the program carefully to determine any conditions that should be added. The following flowchart adds any conditions for the actions that take place during each part of the program.


Step 5:
Add Alarm or Monitoring Operations

Many people are tempted to add alarm or monitoring operations earlier in the flowchart design process. It is almost always easier to add them last because then you know how they should affect the main part of the program. The following flowchart adds an operation that monitors for the conditions that will stop the press.


## Step 6: <br> Determine Stage <br> Numbering

You can number the stages any way you would like, but it's usually best to follow some type of sequence that matches the flow of the program. This makes it much easier to understand. There are a few guidelines we have used to determine the best numbering sequence. You don't have to follow these guidelines, but they may help. You can typically find these types of operations in any program.

- Sequential Operations - a certain sequence of events, one after the other. This is usually the main part of the program. It's usually best to number these first. For example, you may want to always number these stages from 0-127 (octal).
- Independent Operations - these operations usually only perform one task, such as activating a motor or turning on a horn. For example, you may want to number all independent operations starting from 130-147 (octal).
- Alarm and Monitoring Operations - These operations usually monitor the main parts of the program. Since you may want to reset parts of the program during an alarm condition, it is usually best to number these last. This way you can use one Reset (RST) instruction to reset almost the entire program. Use stages 150-177 for alarming and monitoring stages.
These guidelines are especially helpful if you have many different programs. By using a standard numbering scheme, you always know where to look for the various types of operations.
The example shows how we assigned numbers for the example press. Notice we've also made a separate stage for the clamp. This was not an absolute requirement because there are several ways you could have done this. We just did it to show you an example of an independent operation.


Step 7:
Assign I/O Addresses

The final step before you enter the program is to assign the I/O addresses and the destinations for any Jump, Set, or Reset instructions.


Step 8:
Enter the Program

The following diagram shows how the program would look when viewed as a ladder program.


This diagram shows how a portion of the program would look when viewed as a Stage Diagram in DirectSOFT.


## Instruction Set

In This Chapter. . . .

- Boolean Instructions
- Comparative Boolean Instructions
- Timer, Counter, and Shift Register Instructions
- Accumulator Load and Output Instructions
- Accumulator Logic Instructions
- Math Instructions
— Bit Operation Instructions
- Number Conversion Instructions
- Program Control Instructions
- Network Instructions
- Message Instructions


## Introduction

The DL305 CPUs offer a wide variety of instructions to perform many different types of operations. This chapter shows you how to use these individual instructions. The following table provides a quick reference listing of the instruction mnemonic and the page(s) defining the instruction. (The mnemonics are very similar to the instruction names and should be easy to become familiar with in a short time.) For example STR NOT (Comparative) is the mnemonic for Store Not Equal. Each instruction definition will show in parentheses the keystrokes used to enter the instruction.
There are two ways to quickly find the instruction you need.

- If you know the instruction category (Boolean, Comparative Boolean, etc.) just use the header at the top of the page to find the pages that discuss the instructions in that category.
- If you know the individual instruction mnemonic, use the following table to find the page that discusses the instruction.
The DL330 and DL340 instructions sets are very similar. However, the DL330P instruction set has several differences. Some of the instructions in this chapter will be labeled "DL330/DL340 Only." There may be an equivalent instruction for the DL330P CPU, but it may also work slightly differently. The DL330P instructions that operate differently from these instructions are discussed in Chapter 12. Make sure you review the instructions carefully to make sure you can use the instruction with your CPU.

| Instruction | Page |
| :---: | :---: |
| ADD (F71) | 11-34 |
| AND | 11-10 |
| AND (Comparative) | 11-21 |
| AND NOT | 11-10 |
| AND NOT (Comparative) | 11-21 |
| AND NOT T/C | 11-11, 11-12 |
| AND STR | 11-13 |
| AND T/C | 11-11, 11-12 |
| BCD (F86) | 11-48 |
| BIN (F85) | 11-47 |
| CMP (F70) | 11-32 |
| CNT | 11-23 |
| DAND (F75) | 11-30 |
| DECO (F82) | 11-46 |
| DIV (F74) | 11-40 |
| DOR (F76) | 11-31 |
| DOUT (F60) | 11-25 |
| DOUT1 (F61) | 11-26 |
| DOUT2 (F62) | 11-27 |
| DOUT3 (F63) | 11-28 |
| DOUT5 (F65) | 11-29 |
| DSTR (F50) | 11-25 |
| DSTR1 (F51) | 11-26 |
| DSTR2 (F52) | 11-27 |
| DSTR3 (F53) | 11-28 |
| DSTR5 (F55) | 11-29 |
| ENCO (F83) | 11-44 |
| FAULT (F20) | 11-56 |


| Instruction | Page |
| :--- | :--- |
| INV F84) | $11-49$ |
| MCR | $11-50$ |
| MCS | $11-50$ |
| MUL | $11-38$ |
| OR | $11-7$ |
| OR (Comparative) | $11-20$ |
| OR NOT | $11-7$ |
| OR NOT (Comparative) | $11-20$ |
| OR NOT T/C | $11-8,11-9$ |
| OR STR | $11-13$ |
| OR T/C | $11-8,11-9$ |
| OUT | $11-15$ |
| RST | $11-16$ |
| RX | $11-52$ |
| SET | $11-16$ |
| SET OUT | $11-17$ |
| SET OUT RST | $11-18$ |
| SHFL | (F82) |
| SHFR | $11-42$ |
| SR | (F80) |
| STR | $11-43$ |
| STR (Comparative) | $11-24$ |
| STR NOT | $11-4$ |
| STR NOT (Comparative) | $11-19$ |
| STR NOT T/C | $11-5,11-6$ |
| STR T/C | $11-5,11-6$ |
| SUB | $11-36$ |
| TMR | $11-22$ |
| WX | (F72) |

NOTE: See Chapter 12 for RLL ${ }^{P L U S}$ instructions.

## Boolean Instructions

The Store instruction begins a new rung or additional branch in a rung with a normally open contact. Status of the contact will be the same state as the associated image register point or memory location.

The Store Not instruction begins a new rung or additional branch in a rung with a normally closed contact. Status of the contact will be opposite the state of the associated image register point or memory location.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Inputs / Outputs | $000-167$ | $000-177$ | $000-167$ |
|  | $700-767$ | $700-767$ | $700-767$ |
| Control Relays | $160-373$ | $160-373$ | $160-174$ |
|  |  | $1000-1067$ | $200-77$ |
| Special Control Relays | $374-377$ | $374-377$ |  |
|  | $770-777$ | $1074-1077$ | $775-1777$ |
|  |  | $400-577$ |  |
| Shift Register Bits | $400-577$ |  | -- |

In the following Store example, when input 000 is on, output 010 will energize.


In the following Store Not example, when input 000 is off output 010 will energize.

Handheld Programmer Keystrokes

| STR | NOT | SHF | 0 | ENT |
| :--- | :--- | :--- | :--- | :--- | :--- |
| OUT | SHF | 1 | 0 | ENT |

Store Timer

Store Not Timer (STR NOT TMR) DL330/340 Only

The Store Timer instruction begins a new rung or additional branch in a rung with a normally open timer contact. The timer contact T aaa will be on when the timer current value is $\geq$ the preset value of the associated timer.

The Store Not Timer instruction begins a new rung or additional branch in a rung with a normally closed timer contact. The timer contact $T$ aaa will be on when the timer current value is < the preset value of the associated timer.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range* |  |
| :--- | :---: | :---: | :---: | :---: |
|  | aaa | aaa | aaa |  |
| Timer | T | $600-677$ | $600-677$ | -- |

* See Chapter12 for similar RLLPLUS instructions

In the following Store Timer example, when the current value in timer 600 is $\geq$ the preset value output 017 will energize.

Handheld Programmer Keystrokes


In the following Store Not Timer example, when the current value in timer 600 is < the preset value output 017 will energize.


Handheld Programmer Keystrokes
STR NOT TMR SHF $6 \square 0 \square 0 \square$ ENT OUT SHF 1,7 ENT


## Store Counter (STR CNT) DL330/340 Only

Store Not Counter (STR NOT CNT) DL330/340 Only

The Store Counter instruction begins a new rung or additional branch in a rung with a normally open counter contact. The counter contact $C$ aaa will be on when the counter current value $\geq$ the preset value of the associated counter.

The Store Not Counter instruction begins a new rung or additional branch in a rung with a normally closed counter contact. The counter contact $C$ aaa will be on when the counter current value is < the preset value of the associated counter.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range* |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Counter | C | $600-677$ | $600-677$ |

* See Chapter 12 for similar RLLPLUS instructions

In the following Store Counter example, when the current value in counter 602 is $\geq$ the preset value output 015 will energize.


Handheld Programmer Keystrokes


In the following Store Not Counter example, when the current value in counter 602 is < the preset value output 015 will energize.


Or (OR)

Or Not (OR NOT)

The Or instruction logically ors a normally open contact in parallel with another contact in a rung. The status of the contact will be the same state as the associated image register point or memory location.

The Or Not instruction logically ors a normally closed contact in parallel with another contact in a rung. The status of the contact will be opposite the state of the associated image register point or memory location.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Inputs / Outputs | $000-167$ |  |  |
|  | $700-767$ | $000-177$ |  |
|  | $700-767$ | $000-167$ |  |
| Control Relays | $160-373$ | $160-373$ |  |
|  |  | $1000-1067$ | $160-174$ |
|  |  | $374-377$ | $374-377$ |
| Special Control Relays | $770-777$ | $1074-1077$ | $175-177$ |
|  |  | $400-577$ | $770-777$ |
| Shift Register Bits | $400-577$ |  | -- |

In the following Or example, when input 000 or 001 is on output 010 will energize.


Handheld Programmer Keystrokes


In the following Or Not example, when input 000 is on or 001 is off output 010 will energize.


## Or Timer (OR TMR) DL330/340 Only

Or Not Timer (OR NOT TMR) DL330/340 Only

The Or Timer instruction logically ors a normally open timer contact in parallel with another contact in a rung. The timer contact $T$ aaa will be on when the timer current value is $\geq$ the preset value of the associated timer.

The Or Not Timer instruction logically ors a normally closed timer contact in parallel with another contact in a rung. The timer contact T aaa will be on when the timer current value is < the preset value of the associated timer.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range* |  |
| :--- | :---: | :---: | :---: | :---: |
|  | aaa | aaa | aaa |  |
| Timer | T | $600-677$ | $600-677$ | -- |

* See Chapter 12 for similar RLLPLUS instructions

In the following Or Timer example, when input 000 is on or the current value in T600 is $\geq$ the preset value output 010 will energize.


In the following Or Not Timer example, when input 000 is on or the current value in T600 is < the preset value output 010 will energize.


Handheld Programmer Keystrokes


## Or Counter (OR CNT) DL330/340 Only

Or Not Counter (OR NOT CNT) DL330/340 Only

The Or Counter instruction logically ors a normally open counter contact in parallel with another contact in a rung. The counter contact $C$ aaa will be on when the counter current value is $\geq$ the preset value of the associated counter.


The Or Not Counter instruction logically ors a normally closed counter contact in parallel with another contact in a rung. The counter contact $C$ aaa will be on when the counter current value is < the preset value of the associated counter.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range* $^{*}$ |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Counter | C | $600-677$ | $600-677$ |

* See Chapter 12 for similar RLLPLUS instructions

In the following Or Counter example, when input 007 is on or the current value in C610 is $\geq$ the preset value output 025 will energize.


Handheld Programmer Keystrokes


In the following Or Not Counter example, when input location 007 is on or the current value in C 610 is < the preset value output 025 will energize.


And
(AND)

And Not
(AND NOT)

The And instruction logically ands a normally open contact in series with another contact in a rung. The status of the contact will be the same state as the associated image register point or memory location.

The And Not instruction logically ands a normally closed contact in series with another contact in a rung. The status of the contact will be opposite the state of the associated image register point or memory location.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Inputs / Outputs | $000-167$ | $700-767$ | $700-767$ |

In the following And example, when input 000 and 001 is on output 010 will energize.
DirectSOFT Display Handheld Programmer Keystrokes


In the following And Not example, when input 000 is on and 001 is off output 010 will energize.

DirectSOFT Display


Handheld Programmer Keystrokes


And Timer (AND TMR) DL330/340 Only

And Not Timer (AND NOT TMR) DL330/340 Only

The And Timer instruction logically ands a normally open timer contact in series with another contact in a rung. The timer contact $T$ aaa will be on when the timer current value $\geq$ the preset value of the associated timer.

The And Not Timer instruction logically ands a normally closed timer contact in series with another contact in a rung. The timer contact $T$ aaa will be on when the timer current value is < the preset value of the associated timer.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range* |  |
| :--- | :---: | :---: | :---: | :---: |
|  | aaa | aaa | aaa |  |
| Timer | T | $600-677$ | $600-677$ | -- |

* See Chapter 12 for similar RLLPLUS instructions

In the following And Timer example, when input 000 is on and the current value in timer 604 is $\geq$ the preset value output 050 will energize.


In the following And Not Timer example, when input 000 is on and the current value in timer 604 is < the preset value output 050 will energize.


## And Counter (AND CNT) DL330/340 Only

And Not Counter (AND NOT CNT) DL330/340 Only

The And Counter instruction logically ands a normally open counter contact in series with another contact in a rung. The counter contact $C$ aaa will be on when the counter current value is $\geq$ the preset value of the associated counter.

The And Not Counter instruction logically ands a normally closed counter contact in series with another contact in a rung. The counter contact $C$ aaa will be on when the counter current value is < the preset value of the associated counter.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Counter | C | $600-677$ | $600-677$ |

* See Chapter 12 for similar RLLPLUS instructions

In the following And Counter example, when input 002 is on and the current value in counter 602 is $\geq$ the preset value output 050 will energize.


Handheld Programmer Keystrokes


In the following And Not Counter example, when input 002 is on and the current value in counter 602 is < the preset value output 050 will energize.

DirectSOFT Display


Handheld Programmer Keystrokes


And Store
(AND STR)

Or Store (OR STR)

The And Store instruction logically ands two branches of a rung in series. Both branches must begin with the Store instruction.


The Or Store instruction logically ors two branches of a rung in parallel. Both branches must begin with the Store instruction.


In the following And Store example, the branch consisting of contacts 000 and 002 have been anded with the branch consisting of contacts 001 and 003.


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |
| :---: | :---: | :---: | :---: |
| OR | SHF | 2 | ENT |
| STR | SHF | 1 | ENT |
| OR | SHF | ENT |  |
| AND | STR | ENT | ENT |
| OUT | SHF | 1 | 0 |

In the following Or Store example, the branch consisting of 000 and 001 have been ored with the branch consisting of 002 and 003.


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |
| :---: | :---: | :---: | :---: |
| AND | SHF | 1 | ENT |
| STR | SHF | 2 | ENT |
| AND | SHF | 3 | ENT |
| OR | STR | ENT |  |
| OUT | SHF | 1 | 0 |

There are limits to what you can enter with these simple boolean instructions. This is because the DL305 CPUs use an 8 -level stack to evaluate the various logic elements. The stack is a temporary storage area that helps solve the logic for the rung. Each time you enter a Store instruction, the instruction is placed on the top of the stack. Any other instructions on the stack are pushed down a level. The And, Or, And Store, and Or Store instructions combine levels of the stack when they are encountered. Since the stack is only eight levels, an error will occur if the CPU encounters a rung that uses more than the eight levels of the stack.
The following example shows how the stack is used to solve simple boolean logic.


1 STR 000

| 1 | STR 000 |
| :--- | :--- |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |

3 STR 002

| 1 | STR 002 |
| :--- | :--- |
| 2 | STR 001 |
| 3 | STR 000 |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |

4 AND 003

| 1 | 002 AND 003 |
| :--- | :--- |
| 2 | STR 001 |
| 3 | STR 000 |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |

5 ORSTR

| 1 | 001 OR (002 AND 003) |
| :--- | :--- |
| 2 | STR 000 |
| 3 |  |

$\cdot$

| 8 |  |
| :--- | :--- |

6 AND 004

| 1 | 004 AND [001 OR (002 AND 003)] |
| :--- | :--- |
| 2 | STR 000 |
| 3 |  |

$\stackrel{\rightharpoonup}{\bullet}$

| 8 |  |
| :--- | :--- |

7 OR 005

| 1 | 005 OR 004 AND [001 OR (002 AND 003)] |
| :--- | :--- |
| 2 | STR 000 |
| 3 |  |

8

ANDSTR

| 1 | 000 AND (005 OR 004) AND [001 OR (002 AND 003)] |
| :--- | :--- |
| 2 |  |
| 3 |  |

- 

Out (OUT)

The Out instruction reflects the status of the rung (on/off) and outputs the discrete (on/off) state to the specified image register point or memory location. Multiple Out instructions referencing the same discrete location should not be used because only the last Out instruction in the program will control the physical output point.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Outputs | $000-167$ | $000-177$ | $000-167$ |
|  | $700-767$ | $700-767$ | $700-767$ |
| Control Relays | $160-373$ | $160-373$ | $160-174$ |
|  |  | $1000-1067$ | $200-77$ |
| Shift Register Bits | $400-577$ | $400-577$ | -- |

In the following Out example, when input 000 is on output 010 will energize.


In the following Out example, two Out instructions using output 10 are used in the program. The status of output 010 being controlled by input 001 will override the instance of output 010 being controlled by input 000 . The physical output would always be controlled by input 001 .


## Set

(SET)
DL330/340 Only

Reset
(RST)
DL330/340 Only

The Set instruction sets or turns on an output. Once the output is set it will remain on until it is reset using the Reset instruction. It is not necessary for the input controlling the Set instruction to remain on. The Set instruction is sometimes known as a latch. The Reset instruction is used to reset the output.

The Reset instruction resets or turns off an output. Once the output is reset it is not necessary for the input to remain on. The Reset instruction is sometimes known as an unlatch instruction.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range* |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Outputs | $000-167$ |  |  |
| $700-767$ | $000-177$ |  |  |
| $700-767$ | -- |  |  |
| Control Relays | $160-373$ | $160-373$ |  |
|  |  | $4000-1067$ | -- |
| Shift Register Bits | $400-577$ | -- |  |

* See Chapter 12 for similar RLLPLUS instructions

In the following Set example, when input 000 is on output 010 will be energized.
DirectSOFT Display

$|$| 000 |
| :--- | :--- |

In the following Reset example, when input 001 is on output 010 will de-energize.

DirectSOFT Display


Handheld Programmer Keystrokes


Handheld Programmer Keystrokes


Set Out
(SET OUT)

The Set Out instruction reflects the status of the rung (on/off) and outputs the discrete (on/off) state to the specified image register location. This instruction is similar to the Out instruction except the output disable coil (special relay 376) will not override and disable the output. Multiple Set Out instructions referencing the same discrete location should not be used because only the last Set Out instruction in the program will control the physical output point.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Outputs | $000-167$ | $000-177$ | $000-167$ |
|  | $700-767$ | $700-767$ | $700-767$ |

In the following Set Out example, when input location 000 is on output 020 will energize. The output disable coil (special relay 376) will not override this output coil.


## Set Out Reset (SET OUT RST)

The Set Out Reset instruction is typically known as a one shot. When the input logic produces an off to on transition the output will turn on for one CPU scan.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Outputs | $000-167$ | $000-177$ | $000-167$ |
|  | $700-767$ | $700-767$ | $700-767$ |
| Control Relays | $160-373$ | $160-373$ | $160-174$ |
|  |  | $1000-1067$ | $200-77$ |

In the following Set Out Reset example, when input 007 transitions from off to on, control relay 160 will energize for the remainder of the CPU scan.

DirectSOFT Display

$|$| 007 | 160 |
| :---: | :---: |
| $\mid$ |  |
| (OUT RST) |  |

Handheld Programmer Keystrokes

| STR | SHF 7 |
| :--- | :--- |



## Comparative Boolean Instructions

Store If Equal (STR)
DL330/DL340 Only

The Store If Equal instruction begins a new rung or additional branch in a rung with a normally open comparative counter contact. The contact will be on if the specified counter CT aaa $=\mathrm{B}$ bbbb.


CT
The Store Not If Equal instruction begins a new rung or additional branch in a rung with a normally closed comparative counter contact. The contact will be on if the specified counter CT aaa $\neq B$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range* |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | CT | $600-677$ | -- | $600-677$ | -- | -- |  |
| Data registers | R | -- | $400-577$ | -- | $400-577$ |  |  |
| $700-777$ | -- | - |  |  |  |  |  |
| Constant | K | -- | $0-9999$ | -- | $0-9999$ | -- |  |

* See Chapter 12 for similar RLLPLUS instructions

In the following Store If Equal example, when CT600 $=2510$ output 012 will energize.


In the following Store Not If Equal example, when CT600 is $\neq$ the value in R400 output 020 will energize.

DirectSOFT Display


Handheld Programmer Keystrokes


## Or If Equal (OR) DL330/DL340 Only

Or Not If Equal (OR NOT) DL330/DL340 Only

The Or If Equal instruction connects a normally open comparative counter contact in parallel with another contact. The contact will be on if the specified counter CT aaa $=\mathrm{B}$ bbbb.


The Or Not If Equal instruction connects a normally closed comparative counter contact in parallel with another contact. The contact will be on if the specified counter CT aaa $\neq \mathrm{B}$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range* |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | CT | $600-677$ | -- | $600-677$ | -- | -- |  |
| Data registers | R | -- | $400-577$ | -- | $400-577$ |  |  |
| Constant | K | -- | $0-9999$ | -- | -- | - |  |

* See Chapter 12 for similar RLLPLUS instructions

In the following Or If Equal example, when input contact 001 is on or CT600 $=2510$ output 012 will energize.


In the following Or Not If Equal example, when input contact 001 is on or CT600 $\neq$ the value in R400 output 012 will energize.

DirectSOFT Display


And If Equal
(AND)
DL330/DL340 Only

The And If Equal instruction connects a normally open comparative counter contact in series with another contact. The contact will be on if the specified counter CT aaa $=\mathrm{B}$ bbbb.

And Not If Equal (AND NOT) DL330/DL340 Only


The And Not If Equal instruction connects a normally closed comparative counter contact in series with another contact. The contact will be on if the specified counter CT aaa $\neq \mathrm{B}$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range* |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | CT | $600-677$ | -- | $600-677$ | -- | -- | -- |
| Data registers | R | -- | $400-577$ | -- | $400-577$ | -- | -- |
| Constant | K | -- | $0-9999$ | -- | $0-9999$ | -- | -- |

* See Chapter 12 for similar RLLPLUS instructions

In the following And If Equal example, when input contact 001 is on and CT600 = 2510 the contact will turn on and output 012 will energize.
DirectSOFT Display
Handheld Programmer Keystrokes

| STR | SHF | 1 | ENT |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: |
| AND | SHF | 6 | 0 | 0 | ENT |
| SHF | 2 | 5 | 1 | 0 | ENT |
| OUT | SHF | 0 | 1 | 2 | ENT |

In the following And Not If Equal example, when input contact 001 is on and CT600 $\neq$ the value in R400 output 012 will energize.


## Timer, Counter, and Shift Register Instructions

Timer
(TMR)
DL330/DL340 Only

See Errata Sheet at the beginning of this file. The counters and timers both time in Decimal and not in BCD. Presets for both are also interpreted as decimal data and not as BCD.

The Timer instruction provides a single input timer with a 0.1 second increment (0-999.9 seconds) in the normal operating mode, or a 0.01 second increment ( $0-99.99$ seconds) in the fast timer mode when relay 770 is turned on. The timer will time up to 9999 and stop. It will reset to zero when the input is turned off. The discrete bit associated with the timer will be on when the current value is equal to or greater than the preset value.

| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Timers | T | 600-677 | -- | 600-677 | -- | -- | -- |
| Data registers | R | -- | 400-577 | -- | $\begin{aligned} & 400-577 \\ & 700-777 \end{aligned}$ | -- | -- |
| Constant | K | -- | 0-9999 | -- | 0-9999 | - | -- |

* See Chapter 12 for similar RLLPLUS instructions

In the following Timer example, timer 600 will begin timing up when input 000 turns on. The timer bit associated with timer 600 will turn on when the current value in timer 600 is $\geq$ the preset value K30 ( 3 seconds). When input 000 turns off the timer discrete bit and current value are reset.

DirectSOFT Display



Counter
(CNT)
DL330/DL340 Only
See Errata Sheet at the beginning of this file. The counters and timers both time in Decimal and not in BCD. Presets for both are also interpreted as decimal data and not as BCD.

The Counter instruction provides a counter with a count and reset input. The range of this counter is 0-9999 and it will increment when the count input transitions from off to on. The counter is reset to 0 when you turn on the reset input. The counter bit associated with the counter will turn on when the current value is equal to or greater than the preset value.

| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | CT | 600-677 | -- | 600-677 | -- | -- | -- |
| Data registers | R | -- | 400-577 | -- | $\begin{aligned} & 400-577 \\ & 700-777 \end{aligned}$ | -- | -- |
| Constant | K | -- | 0-9999 | -- | 0-9999 | -- | -- |

* See Chapter 12 for similar RLLPLUS instructions

In the following Counter example, counter 604 will increment by one count when input 000 transitions from off to on. When input contact 001 is turned on the counter will reset to zero. The counter bit associated with counter 604 will turn on when the current value in counter 604 is $\geq$ the preset constant value K3 (3).


Shift Register<br>(SR)<br>DL330/340 Only

The Shift Register instruction shifts data through a predefined number of shift register bits. There are 128 bits allocated for use in shift registers. There is no limit to the number of shift registers which can be used in a program, however the total number of bits used cannot exceed 128. The Shift Register has three contacts.

- Data - determines the value (1 or 0) that will enter the register
- Clock - shifts the bits one position on each off to on transition
- Reset -resets the Shift Register to all zeros.


With each off to on transition of the clock input, the bits which make up the shift register block are shifted by one bit position and the status of the data input is shifted into the starting bit position in the block. The direction of the shift depends on the entry in the From and To fields. From 400 to 407 would define a block of eight bits to be shifted from bit 400 to bit 407 . From 407 to 400 would also define a block of eight bits, but would shift from bit 407 to bit 400 . The maximum size of the shift register block is limited to 128 bits. There is no minimum block size.

| Operand Data Type | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: |
|  | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Shift Register Bits | $400-577$ | $400-577$ | $400-577$ | $400-577$ | -- | -- |

DirectSOFT Display


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |  |
| :--- | :--- | :--- | :--- | :--- |
| STR | SHF | 1 | ENT |  |
| STR | SHF | 2 | ENT |  |
| STR | SHF |  |  |  |
| SR | SHF | 4 | 0 | 0 |
| SHF | 4 | 1 | 1 | 7 |
|  |  | ENT |  |  |

Inputs on Successive Scans Shift Register Bits

| Data | clock | Reset |  |
| :---: | :---: | :---: | :---: |
| 1 | 1 | 0 |  |
| 0 | 1 | 0 | - पПШПППППП |
| 0 | 1 | 0 |  |
| 1 | 1 | 0 |  |
| 0 | 1 | 0 |  |
| 0 | 0 | 1 |  |

- indicates on $\square$ - indicates off


## Accumulator Load and Output Instructions

Data Store DSTR (F50)

## Data Out DOUT (F60)

The Data Store (F50) is a 16-bit instruction that loads the value of a 16-bit register, two consecutive 8-bit registers (specify starting location), or a 4-digit $B C D$ value into the accumulator.


The Data Out (F60) is a 16-bit instruction that copies the 16 -bit value in the accumulator to a 16-bit reference or two consecutive 8 -bit registers (specify starting location).


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> $700-777$ | $0000-9999$ |
| *Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ |  |

* A constant is not a valid data type for the DOUT (F60) instruction.

In the following example, when input 000 is on the value (7502) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The value in the accumulator is output to data registers R404 and R405 using the Data Out (F60) instruction.

DirectSOFT Display



Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |  |
| :---: | :---: | :---: | :---: | :---: |
| F | 5 | 0 | ENT |  |
| R | 4 | 0 | 2 | ENT |
| F | 6 | 0 | ENT |  |
| R | 4 | 0 | 4 | ENT |

In the following example, when input 001 is on the BCD constant value K7502 is loaded into the accumulator using the Data Store (F50) instruction. The value in the accumulator is output to data registers R404 and R405 using the Data Out (F60) instruction.


| Handheld Programmer Keystrokes |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: |
| STR | SHF | 1 | ENT- |  |  |
| F | 5 | 0 | ENT |  |  |
| SHF | 7 | 5 | 0 | 2 | ENT |
| F | 6 | 0 | ENT |  |  |
| R | 4 | 0 | 4 | ENT |  |

Data Store 1 DSTR (F51)

Data Out 1 DOUT (F61)

The Data Store 1 (F51) is an 8-bit instruction that loads the value from a specified 8-bit register into the lower 8 bits of the accumulator. The upper 8 bits of the accumulator are set to zero.


The Data Out 1 (F61) is an 8-bit instruction that copies the value in the lower 8 bits of the accumulator to a specified 8-bit register.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  |  | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $040-056$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | -- | $040-056$ | -- |
| Stages | R | $400-577$ | $400-577$ <br> $700-777$ | $400-116$ |
| Data Registers |  |  |  |  |

In the following example, when input 000 is on the value (89) in R410 is loaded into the lower 8 bits of the accumulator using the Data Store 1 (F51) instruction. The value in the least significant 8 bits of the accumulator is output to data register R500 using the Data Out 1 (F61) instruction.

DirectSOFT Display


| STR | SHF | 0 | ENT |
| :---: | :---: | :---: | :---: |
| F | 5 | 1 | ENT |
| R | 4 | 1 | 1 |
| F | 0 | 0 | ENT |
| R | 6 | 1 | ENT |
| 5 | 0 | 0 |  |

Data Store 2 DSTR (F52)

Data Out 2 DOUT (F62)

The Data Store 2 (F52) is a 4-bit instruction that loads the value of the most significant 4 bits of a specified 8-bit register into the least significant 4 bits of the accumulator. The remaining 12 bits of the accumulator are set to zero.

The Data Out 2 (F62) is a 4-bit instruction that copies the value in the least significant 4 bits of the accumulator into the most significant 4 bits of a specified 8-bit register. The lower 4 bits of the register are not altered .


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: |
|  |  | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-27$ |
| Shift Registers | R | $040-056$ | $040-56$ | -- |
| Stages | R | -- | -- | $400-116$ |
| Data Registers | R | $400-577$ | $400-577$ |  |

In the following example, when input 000 is on the most significant 4 bits of R1 are loaded into the lower 4 bits of the accumulator using the Data Store 2 (F52) instruction. The value in the least significant 4 bits of the accumulator is output to most significant 4 bits of data register R400 using the Data Out 2 (F62) instruction.


Handheld Programmer Keystrokes

accumulator to the upper 4
bits or registers R400

Data Store 3
DSTR (F53)

Data Out 3 DOUT (F63)

The Data Store 3 (F53) is a 4-bit instruction that loads the value of the least significant 4 bits of a specified 8 -bit register into the least significant 4 bits of the accumulator. The upper 12 bits of the accumulator are set to zero.

The Data Out 3 (F63) is a 4-bit instruction that copies the value in the least significant 4 bits of the accumulator to the least significant 4 bits of a specified 8 bit register. The upper 4 bits of the register are not altered.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: |
|  |  | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Data Registers | $R$ | $400-577$ | $400-577$ |  |

In the following example, when input 000 is on the least significant 4 bits of R005 are loaded into the accumulator using the Data Store 3 (F53) instruction. The data in the least significant 4 bits of the accumulator is output to the least significant 4 bits of R016 using the Data Out 3 (F63) instruction.


Data Store 5
DSTR (F55)

Data Out 5 DOUT (F65)

The Data Store 5 (F55) is a 16 -bit instruction that loads the value of 16 image register locations for a specified 16 point input module into the accumulator.

The Data Out 5 (F65) is a 16-bit instruction that outputs the 16 bit value in the accumulator to the image register of a specified 16 point output module.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :---: | :---: | :---: | :---: |
|  | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |

In the following example, when input 000 is on the binary status of a 16 point I/O module in slot 1 (R000 and R010) is loaded into the accumulator using the Data Store 5 (F55) instruction. The value in the accumulator is copied to I/O register locations in slot 2 (R001 and R011) using the Data Out 5 (F65) instruction.


## Accumulator Logic Instructions

Data And<br>DAND (F75)

The Data And (F75) is a 16-bit instruction that logical ands the value in a 16-bit reference, two consecutive 8-bit registers (specify starting location), or a 4-digit BCD constant with the value in the accumulator. The result resides in the
 accumulator.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $700-777$ | $0000-9999$ |
| Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ |  |

In the following example, when input 000 is on the value(6489) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The data in the accumulator is logically anded with the constant K4107 with the result residing in the accumulator. The value in the accumulator is output to data register R404 and R405 using the Data Out (F60) instruction.



The Data Or (F76) is a 16-bit instruction that logically ors the value in a 16-bit reference, two consecutive 8-bit registers, (specify starting location) or a 4-digit BCD constant with the value in the accumulator. The result resides in the accumulator.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $700-777$ | $0000-9999$ |
| Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ |  |

In the following example, when input 000 is on the value (6481) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The data in the accumulator is logically ored with the constant K4102 with the result residing in the accumulator. The value in the accumulator is output to data registers R404 and R405 using the Data Out (F60) instruction.


Compare
CMP (F70)

The Compare (F70) is a 16-bit instruction that compares the value in a 16-bit reference, two consecutive 8-bit registers (specify starting location), or a 4-digit BCD against the value in the accumulator. Discrete bit flags are used to indicate if the result of the comparison was greater than, equal to, or less than the value in the accumulator.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-27$ |
| Shift Registers | R | $040-056$ | $040-56$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $700-577$ |  |
| Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ | $0000-9999$ |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 772 | Will be on if the accumulator value is greater than the compare value |
| 773 | Will be on if the accumulator value is equal to the compare value |
| 774 | Will be on if the accumulator value is less than the compare value |

In the following example, when input 000 is on the value (2132) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The data in the accumulator is compared to value in data registers R404 and R405 using the Compare (F70) instruction. Discrete status flag 774 is used to indicate if the accumulator is less than the compare value in this example.

DirectSOFT Display


Compare the value in the accumulator with the value in


Flag status after Compare execution

## 772 off

73 off
774 on

Handheld Programmer Keystrokes


| STR | SHF | 0 | ENT |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: |
| F | 5 | 0 | ENT |  |  |
| R | 4 | 0 | 2 | ENT |  |
| F | 7 | 0 | ENT |  |  |
| R | 4 | 0 | 4 | ENT |  |
| STR | SHF | 0 | ENT |  |  |
| AND | SHF | 7 | 7 | 4 | ENT |
| OUT | SHF | 1 | 6 | 0 | ENT |



NOTE: Input 000 has been used to interlock output 160. This is done since an earlier comparison could result in status flag 774 coming on when this particular comparison is not being executed thereby providing the opportunity for an unexpected output signal on output 160.
It is a common mistake to just use the status flags without interlocking to control outputs in a program but, status flags 772-774 can change several times during the same scan. Just as you should not use the status flags by themselves to control outputs, you also should not monitor status flags within the program. Instead you should monitor the interlocked outputs controlled by the status flags.

## Math Instructions

The Add (F71) is a 16-bit instruction that adds the value of a 16 bit reference, two consecutive 8-bit registers (specify starting location), or a 4-digit BCD value with the value in the accumulator. The result resides in the accumulator.
 Discrete bit flags are used to indicate if the result had a carry digit or if the result was was zero.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> $700-777$ | $0000-9999$ |
| Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ |  |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 775 | Will be on if the operation results in a carry |
| 776 | Will be on if the result is 0 |

In the following example, when input 000 is on the value (3619) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The Add instruction (F71) adds the value (2602) in R410 and R411 to the value in the accumulator. The result in the accumulator is then copied to data registers R404 and R405 with the Data Out (F60) instruction.



## Add Example

The following examples demonstrate how the discrete status flags are used to indicate if the result of the add has produced a number which exceeds the capacity of the accumulator. Remember, the accumulator has a 4 digit maximum. When a calculation produces a number larger than 4 digits, part of this number is lost. The following table shows different values being used in the logic example below. Notice how the discrete status flags change.

|  | Registers for DSTR <br> Instruction <br> R401/R400 | Registers for ADD <br> Instruction <br> R403/R402 | Registers for DOUT <br> Instruction <br> R405/R404 | Discrete Status <br> Flag <br> 775 | Discrete Status <br> Flag <br> 776 |
| :--- | :--- | :--- | :--- | :--- | :--- |
| Example 1 | 500 | 400 | 0900 | off | off |
| Example 2 | 5000 | 5000 | 0000 | on | on |
| Example 3 | 5050 | 5000 | 0050 | on | off |

DirectSOFT Display


Add the value in the accumulator with the value in registers R402 and R403


Output the value in the accumulator to registers R404 and R405


NOTE: An input has been used to interlock the outputs on the last two rungs. This is done since an earlier math instruction could result in the status flag coming on when this particular math instruction is not being executed thereby providing the opportunity for an unexpected output signal.

It is a common mistake to just use the status flags without interlocking to control outputs in a program but, the status flags can change several times during the same scan. Just as you should not use the status flags by themselves to control outputs, you also should not monitor status flags within the program. Instead you should monitor the interlocked outputs controlled by the status flags.

## Subtract SUB (F72)

The Subtract (F72) is a 16-bit instruction that subtracts the value in a 16-bit register, two consecutive 8-bit registers (specify starting location), or a 4-digit $B C D$ value from the value in the accumulator. The result resides in the accumulator. Discrete bit flags are used to indicate if the result had a borrow digit or the result was zero.

| Data Type |  | D3-330 Range | D3-340 Range | D3-330P Range |
| :---: | :---: | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $\begin{aligned} & 000-014 \\ & 070-075 \end{aligned}$ | $\begin{aligned} & 000-014 \\ & 070-075 \end{aligned}$ | $\begin{aligned} & 000-014 \\ & 070-075 \end{aligned}$ |
| Control Relays | R | 016-036 | $\begin{aligned} & 016-036 \\ & 100-105 \end{aligned}$ | 016, 020-027 |
| Shift Registers | R | 040-056 | 040-056 | -- |
| Stages | R | -- | -- | 100-116 |
| Timer /Counters (16 bit) | R | 600-677 | 600-677 | 600-677 |
| Data Registers | R | 400-577 | $\begin{aligned} & 400-577 \\ & 700-777 \end{aligned}$ | 400-577 |
| Constant (4-digit BCD) | K | 0000-9999 | 0000-9999 | 0000-9999 |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 775 | Will be on if the result if a borrow digit occurred |
| 776 | Will be on if the result is 0 |

In the following example, when input 000 is on the value (3619) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The constant value K1406 is subtracted from the value in the accumulator using the Subtract (F72) instruction. The result in the accumulator is then copied to data registers R404 and R405 using the Data Out (F60) instruction.


# Subtract Example 

The following examples demonstrate how the discrete status flags are used to indicate if the result of the Subtraction is a 0 or required a borrow digit. The following table shows different values being used in the logic example below. Notice how the discrete status flags change for each example.

|  | Registers for DSTR <br> Instruction <br> R401/R400 | Registers for SUB <br> Instruction <br> R403/R402 | Registers for DOUT <br> Instruction <br> R405/R404 | Discrete Status <br> Flag <br> 775 | Discrete Status <br> Flag <br> 776 |
| :--- | :--- | :--- | :--- | :--- | :--- |
| Example 1 | 6050 | 5000 | 1050 | off | off |
| Example 2 | 7050 | 7050 | 0000 | off | on |
| Example 3 | 5000 | 6000 | $9000^{*}$ | on | off |

*The DL305 cannot process negative numbers. When the number being subtracted from the accumulator is larger than the number in the accumulator, a borrow digit occurs and the subtraction is completed. The value in the accumulator does not represent the difference between the two numbers. To get the difference between the two numbers in Example 3 the result (9000) in the accumulator is subtracted from 0 . The final result is 1000 , the difference between 6000 and 5000 .


NOTE: It is a common mistake to just use the status flags without interlocking to control outputs in a program, but the status flags can change several times during the same scan. Just as you should not use the status flags by themselves to control outputs, you also should not monitor status flags within the program. Instead you should monitor the interlocked outputs controlled by the status flags.

Multiply<br>MUL (F73)

The Multiply (F73) is a 16-bit instruction that multiplies the value in a 16-bit register, two consecutive 8-bit registers, or a 4 -digit BCD value by the value in the accumulator. The least significant four digits of the result are stored in the
 accumulator and the most significant four digits are stored in the auxiliary accumulator (R575 and R577). A discrete bit flag is used to indicate if the result was zero.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> $700-777$ | $0000-9999$ |
| Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ |  |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 776 | Will be on if the result is 0 |

In the following example, when input 000 is on the value (3619) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The data in the accumulator is multiplied with the constant K2 with the result residing in the accumulator and auxiliary accumulator (R576 and R577) using the Multiply (F73) instruction. The value in the accumulator is output to data registers R404 and R405 using the Data Out (F60) instruction.


| Handheld Programmer Keystrokes |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: |
| STR | SHF | 0 | ENT |  |
| F | 5 | 0 | ENT |  |
| R | 4 | 0 | 2 | ENT |
| F | 7 | 3 | ENT |  |
| SHF | 2 | ENT |  |  |
| F | 6 | 0 | ENT |  |
| R | 4 | 0 | 4 | ENT |

Multiply Example The multiply instruction allows you to multiply two 4-digit numbers together. The result is located in the accumulator and the auxiliary accumulator (R576 and R577) when necessary. The accumulator holds the lower 4 digits of the result and the auxiliary accumulator holds the upper 4 digits.

Whenever possible multiplications resulting in more than 4 digits should be avoided since the DL305 instruction set can only manipulate a maximum of two consecutive 8 -bit registers (4 digits) at one time.

If the result of a multiplication is greater than 4 digits, the application program must be written to compensate for the instruction set 4 digit maximum for data manipulation. The example below shows how the auxiliary accumulator is used to store a result with more than 4 digits and how to access the upper 4 digits.

The example below shows how the auxiliary accumulator is used to process numbers larger than 4 digits when the multiplication instruction is used.


| STR | SHF | 0 | ENT |  |
| :---: | :---: | :---: | :---: | :---: |
| F | 5 | 0 | ENT |  |
| R | 4 | 0 | 2 | ENT |
| F | 7 | 3 | ENT |  |
| SHF | 5 | 0 | ENT |  |
| F | 6 | 0 | ENT |  |
| R | 4 | 1 | 2 | ENT |
| F | 5 | 0 | ENT |  |
| R | 5 | 7 | 6 | ENT |
| F | 6 | 0 | ENT |  |
| R | 4 | 1 | 4 | ENT |

Divide<br>DIV (F74)

The Divide (F74) is a 16-bit instruction that divides the value in the accumulator by the value in a 16-bit register, two consecutive 8-bit registers, or a 4-digit BCD value. The integer portion of the result is stored in the accumulator and the decimal fraction is stored in the auxiliary accumulator, R576 and R577. Discrete flags are used to indicate if the dividend or divisor is zero or if only the divisor is zero.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> Constant (4-digit BCD) | K |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 776 | Will be on if the dividend or divisor is zero |
| 777 | Will be on if the divisor is zero |

In the following example, when input 000 is on the value (530) in R402 and R403 is loaded into the accumulator using the Data Store (F50) instruction. The data in the accumulator is divided by the constant 10 (K10). The result in the accumulator and is copied to data registers R404 and R405 using the Data Out (F60) instruction. The remainder is in the auxiliary accumulator (R576 and R577).

DirectSOFT Display


Divide Example The divide instruction allows you to divide the value in the accumulator by 4 digits maximum. The divide instruction uses the accumulator for the integer value of the result and the auxiliary accumulator (R576 and R577) for fraction. The instruction set only allows manipulation on two consecutive registers at a time. For example, if the result was a 4 digit number with a remainder it would have to be treated like two 4-digit numbers in the program. Manipulating numbers over 4 digits should be avoided whenever possible. If it cannot be avoided the application program must be written to compensate for the 4-digit maximum for data manipulation.
The example below shows how the auxiliary accumulator is used to store the fractional portion of the result and how to access the remainder.

DirectSOFT Display


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |  |
| :---: | :---: | :---: | :---: | :---: |
| F | 5 | 0 | ENT |  |
| R | 4 | 0 | 2 | ENT |
| F | 7 | 4 | ENT |  |
| SHF | 3 | 2 | ENT |  |
| F | 6 | 0 | ENT |  |
| R | 4 | 1 | 2 | ENT |
| F | 5 | 0 | ENT |  |
| R | 5 | 7 | 6 | ENT |
| F | 6 | 0 | ENT |  |
| R | 4 | 1 | 4 | ENT |

## Bit Operation Instructions

Shift Left
SHFL (F81)

The Shift Left (F81) is a 16-bit instruction that shifts the value in the accumulator a specified number of bits ( 15 maximum) to the left. Discrete bit flags are used to indicate if a " 1 " was shifted out of the accumulator or if the accumulator equals " 0 " after the shift.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :---: | :---: | :---: | :---: | :---: |
|  | aaaa | aaaa | aaaa |  |
| Constant (4-digit BCD) | K | $1-16$ | $1-16$ | $1-16$ |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 775 | Will be on if a"1" was shifted out of the accumulator. |
| 776 | Will be on if the accumulator equals zero after the shift instruction is executed. |

In the following example, when input 000 is on the value in R000 and R010 is loaded into the accumulator using the Data Store 5 (F55) instruction. The bit pattern in the accumulator is shifted to the left 4 bit positions using the Shift Left (F81) instruction with the result resides in the accumulator. The value in the accumulator is copied to data registers R404 and R405 using the Data Out (F60) instruction.


Shift Right
SHFR (F80)

The Shift Right (F80) is a 16-bit instruction that shifts the value in the accumulator a specified number of bits ( 15 maximum) to the right. Discrete bit flags are used to indicate if a " 1 " was shifted out of the accumulator or if the accumulator equals " 0 " after the shift.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :---: | :---: | :---: | :---: |
|  | aaaa | aaaa | aaaa |
| Constant (4-digit BCD) | K | $1-16$ | $1-16$ |
| $1-16$ |  |  |  |


| Discrete Bit Flags | Description |
| :--- | :--- |
| 775 | Will be on if a"1" was shifted out of the accumulator. |
| 776 | Will be on if the accumulator equals zero after the shift instruction is executed. |

In the following example, when input 000 is on the value in R000 and R010 is loaded into the accumulator using the Data Store 5 (F55) instruction. The bit pattern in the accumulator is shifted 4 bit positions using the Shift Right (F80) instruction and the result resides in the accumulator. The value in the accumulator is copied to data registers R404 and R405 using the Data Out (F60) instruction.


## Number Conversion Instructions

The Encode instruction encodes the accumulator bit position that contains a 1 by returning the corresponding binary representation. If the most significant bit is set to HEX F (decimal 15), the binary value 15 is returned to the accumulator. If the accumulator value is 0000 or 0001 a zero will be returned to the accumulator. If there is more than one bit position set to a "1" the least significant "1" will be encoded. The discrete bit flag 777 is used to indicate if there were multiple 1 s in the accumulator.


| Discrete Bit Flags | Description |
| :--- | :--- |
| 777 | Will be on if there was more than one bit position set to a "1" in the accumulator. |

In the following example, when input 000 is on the 16-bit binary pattern from registers R005 and R015 is loaded in the accumulator by the Data Store 5 (F55) instruction. In this example the $6^{\text {th }}$ bit (BCD 40) is on. When the Encode (F83) instruction executes the accumulator will contain the BCD value of 6 . The lower four bits of the accumulator are copied to the lower four bits of register R404 by the Data Out 3 (F63) instruction.


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |  |
| :---: | :---: | :---: | :---: | :---: |
| F | 5 | 5 | ENT |  |
| R | 5 | ENT |  |  |
| F | 8 | 3 | ENT |  |
| F | 6 | 3 | ENT |  |
| R | 4 | 0 | 4 | ENT |

Decode DECOD (F82)

The Decode instruction decodes a four bit binary number ( $0-\mathrm{F}$ ) in the accumulator and sets the corresponding bit position to a one. If the accumulator contains a HEX F (decimal 15) the most significant bit (bit 15) will be set in the
 accumulator. If the accumulator contains a zero the least significant bit (bit 0) will be set to a one. All other bits in the accumulator will be set to a zero.

In the following example, when 000 is on the binary value of the lower four bits in R016 (5) will be loaded in the accumulator by the Data Store 3 (F53) instruction. The Decode instruction will then translate the value 5 to a 1 in the fifth bit position of the accumulator. The value 20 in the accumulator is copied to data registers R404 and R405 with the Data Out (F60) instruction.


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |
| :---: | :---: | :---: | :---: |
| $\mathrm{F}_{1}$ | 5 | 3 | ENT |
| R | 1 | 1 | 6 |
| F | ENT |  |  |
| F | 8 | 2 | ENT |
| F | 6 | 0 | ENT |
| R | 4 | 0 | 4 | ENT

Binary
BIN (F85)

The Binary (F85) instruction converts a $B C D$ value in the accumulator to the binary/HEX equivalent value. The result of the conversion resides in the accumulator.

In the following example, when input 000 is on the value ( 2571 BCD) in R600 is loaded into the accumulator using the Data Store (F50) instruction. The value in the accumulator is converted to a binary number (HEX OAOB) using the Binary (F85) instruction with the result residing in the accumulator. The value in the accumulator is copied to I/O registers R000 and R010 (which corresponds to I/O points 0-7 and 100-107) with the Data Out 5 (F65) instruction.

Binary Coded
Decimal
BCD (F86)

The Binary Coded Decimal (F86) instruction converts a binary/HEX value in the accumulator to the BCD equivalent. The result of the conversion resides in the accumulator.


In the following example, when input 000 is on the value (HEX OAOA) in R000 and R010 is loaded into the accumulator with the Data Store 5 (F55) instruction. The value in the accumulator is converted to a BCD number (BCD 2570) using the BCD (F86) instruction with the result residing in the accumulator. The value in the accumulator is output to register R600 using the Data Out (F60) instruction.


The Invert instruction generates the one's complement of the number in the accumulator. The result is stored in the accumulator.


In the following example, when input 000 is on the value (AD63) in R000 and R010 is loaded into the accumulator using the Data Store (F55) instruction. The value in the accumulator is inverted with the result residing in the accumulator. The value (HEX 529C) is copied to registers R404 and R405 using the Data Out (F60) instruction.


Copy the value in the
accumulator to registers R404 and R405

## Program Control Instructions

Master Control Set (MCS)
DL330/DL340 Only
Master Control Reset
(MCR)
DL330/DL340 only
Understanding Master Control Relays

The Master Control Set and Master Control Reset instructions are used to provide an additional left power rail which is controllable by a input contact. This is sometimes known as a sub power rail. Any number of rungs of ladder logic can
 be disabled using these instructions.

The Master Control Set (MCS) and Master Control Reset (MCR) instructions allow you to quickly enable (or disable) sections of the RLL program. This provides program control flexibility. The following example shows how the MCS and MCR instructions operate by creating a sub power rail for control logic.


MCS/MCR Example In the following MCS/MCR example logic between the first MCS (A) and the last MCR (B) will function only if input 000 is on. The logic between the second MCS (C) and the next to last MCR (D) will function only if input 010 is on. The last rung is not controlled by either of the MCS coils.


NOTE: When programming the MCS instruction, do not put any parallel coils in the rung with the MCS.

## Network Instructions

Read from Network RX (F952) DL340 Only

The Read from Network instruction is used by the master device on a DirectNET network to read a block of data from another CPU or DirectNET interface module. The function parameters are loaded into the accumulator and the first and second level of the accumulator stack by three additional instructions. Listed
 below are the steps necessary to program the Read from Network function.
Step 1: - Load the slave address (1-90 BCD) into the accumulator with the DSTR instruction. This must always be preceded by 00, so address 20 would be 0020. (Remember, D4-DCM slave device addresses are set with switches that use a hexadecimal format. Make sure you convert this address to the decimal equivalent for use with this instruction.)
Step 2: — Load the number of bytes (1-128 BCD) to be transferred from the network slave station.
Step 3: - Load the octal address for the data register that will be used to store the data obtained from the slave station.
Step 4: - Insert the RX instruction which specifies the starting address in the slave station. If you are getting the data from a DL305 station, just enter the Data Register number. If you are getting the data from a DL205 or DL405 station, enter a constant that corresponds to the memory address. For example, to get the current count for Timer 600 from a DL305 CPU, you would use R600 with the RX instruction. If you wanted to get Counter 0 from a DL405 CPU, you would use a constant of 1000 with the RX instruction. (V1000 stores the current count for Counter 0 in a DL405 CPU.)

NOTE: The DirectNET manual provides further information on the use of the RX and WX network instructions.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> *Constant (4-digit BCD) | K |

* A constant is used to obtain data from a DL205 or DL405 system.

| Discrete Bit Flags | Description |
| :--- | :--- |
| 777 | Parameters are not properly set. Check the slave address, data length, or data <br> address reference. |
| 1074 | Communication port busy. |
| 1075 | Communication error. Data was not transmitted. |

NOTE: See the DL205 or DL405 User's Manuals for a listing of V-memory addresses available with these CPUs. Since the DL305 only supports a 4-digit constant, you will not be able to access the entire V-memory ranges of the DL205 and DL405 CPUs. For example, you could not directly access V40400 stored in a DL405 CPU. If you require data from a range outside the area available with a 4-digit constant (from V0 - V9999) then add a routine to the slave station program that moves this data down into one of the accessible areas.

In the following example, when input 001 is on and the CPU busy relay 1074 (see special relays, p. 8-32) is not on, the RX instruction will access a DL405 CPU that has been assigned station address 20. (Note, the D4-DCM slave station addresses are set with switches that indicate a hexadecimal number. Make sure you determine the decimal equivalent to be used with the first DSTR instruction in the sequence.) Ten consecutive bytes of data (V1400 - V1404) will be read from the slave station and stored in registers R400 - R411. (Remember, the DL205 and DL405 V-memory locations are 16 bits. The DL305 registers are only 8 bits, so you have to use two data registers for each V-memory location.)


Handheld Programmer Keystrokes

| STR | SHF | 1 | ENT |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| AND | NOT | SHF | 1 | 0 | 7 | 4 | ENT |
| F | 5 | 0 | ENT |  |  |  |  |
| SHF | 0 | 0 | 2 | 0 | ENT |  |  |
| F | 5 | 0 | ENT |  |  |  |  |
| SHF | 1 | 0 | ENT |  |  |  |  |
| F | 5 | 0 | ENT |  |  |  |  |
| SHF | 4 | 0 | 0 | ENT |  |  |  |
| F | 9 | 5 | 2 | ENT |  |  |  |
| SHF | 1 | 4 | 0 | 0 | ENT |  |  |

## Write to Network WX (F953) DL340 Only

The Write to Network instruction is used by the master device on a DirectNET network to write a block of data to another station. The function parameters are loaded into the accumulator and the first and second level of the accumulator stack by three additional instructions. Listed below are the steps necessary to program the Write to Network function.
Step 1: - Load the slave address (1-90 BCD) into the accumulator with the DSTR instruction. This must always be preceded by 00, so address 20 would be 0020. (Remember, the D4-DCM slave device addresses are set with switches that use a hexadecimal format. Make sure you convert this address to the decimal equivalent for use with this instruction.)
Step 2: - Load the number of bytes ( $1-128 \mathrm{BCD}$ ) to be transferred to the network slave station.

Step 3: - Load the octal address for the data register that will be used to obtain the data that will be sent to the slave station.

Step 4: - Insert the WX instruction which specifies the starting address in the slave station. If you are sending data to a DL305 station, just enter the Data Register number. If you are sending data to a DL205 or DL405 station, enter a constant that corresponds to the memory address. For example, to send data to Register 500 in a DL305 CPU, you would use R500 with the RX instruction. If you wanted to send data to V1400 in a DL405 CPU, you would use a constant of 1400 with the WX instruction.

NOTE: The DirectNET manual provides further information on the use of the RX and WX network instructions.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-027$ |
| Shift Registers | R | $040-056$ | $040-056$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> $700-777$ | $0000-9999$ |
| *Constant (4-digit BCD) | K | $0000-9999$ | $0000-9999$ |  |

*A constant is used to send data to a DL205 or DL405 system.

| Discrete Bit Flags | Description |
| :--- | :--- |
| 777 | Parameters are not properly set. Check the slave address, data length, or data <br> address reference. |
| 1074 | Communication port busy. |
| 1075 | Communication error. Data was not transmitted. |

NOTE: See the DL205 or DL405 User's Manuals for a listing of V-memory addresses available with these CPUs. Since the DL305 only supports a 4-digit constant, you will not be able to access the entire V-memory ranges of the DL205 and DL405 CPUs. For example, you could not directly access V40400 stored in a DL405 CPU. If you want to send data to a range outside the area available with a 4-digit constant (from V0 - V9999) then add a routine to the slave station program that moves the data from one of the accessible areas into the unavailable locations.

In the following example, when input 001 is on and the CPU busy relay 1074 (see special relays, p. 8-32) is not on, the WX instruction will access a DL405 CPU that has been assigned station address 20. (Note, the D4-DCM slave station addresses are set with switches that indicate a hexadecimal number. Make sure you determine the decimal equivalent to be used with the first DSTR instruction in the sequence.) Ten consecutive bytes of data (R400-R411) will be sent from the DL340 and stored in V-memory locations V1400 - V1404. (Remember, the DL205 and DL405 V-memory locations are 16 bits. The DL305 registers are only 8 bits, so you have to use two data registers for each V-memory location.)

DirectSOFT Display


Handheld Programmer Keystrokes

| STR | SHF | 1 | ENT |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| AND | NOT | SHF | 1 | 0 | 7 | 4 | ENT |
| F | 5 | 0 | ENT |  |  |  |  |
| SHF | 0 | 0 | 2 | 0 | ENT |  |  |
| F | 5 | 0 | ENT |  |  |  |  |
| SHF | 1 | 0 | ENT |  |  |  |  |
| F | 5 | 0 | ENT |  |  |  |  |
| SHF | 4 | 0 | 0 | ENT |  |  |  |
| F | 9 | 5 | 3 | ENT |  |  |  |
| SHF | 1 | 4 | 0 | 0 | ENT |  |  |

## Message Instructions

Fault
FAULT (F20)

The Fault (F20) instruction is used to display a 4-digit BCD constant, 16-bit register, or two consecutive 8 -bit data registers on the handheld programmer or in DirectSOFT. When the fault instruction is executed the number being
 displayed will also be copied into the registers R574 and R575 and the discrete bit flag 771 will be on.

In the following example, when input 000 is on the number 0206 will be displayed on the programming device. This would typically be a user defined error code.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :--- | :--- | :---: | :---: | :---: |
|  | A | aaaa | aaaa | aaaa |
| Inputs / Outputs | R | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ | $000-014$ <br> $070-075$ |
| Control Relays | R | $016-036$ | $016-036$ <br> $100-105$ | $016,020-27$ |
| Shift Registers | R | $040-056$ | $040-56$ | -- |
| Stages | R | -- | -- | $100-116$ |
| Timer /Counters (16 bit) | R | $600-677$ | $600-677$ | $400-577$ |
| Data Registers | R | $400-577$ | $400-577$ <br> $700-777$ | $0000-977$ |
| Constant (4-digit BCD) | K | $0000-9999$ |  |  |

DirectSOFT Display


Handheld Programmer Display


Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |  |
| :---: | :---: | :---: | :---: | :---: |
| F | 2 | 0 | ENT |  |
|  |  |  |  |  |
| SHF | 2 | 0 | 6 | ENT |

## RLL Pus

## Instruction Set

In This Chapter. . . .

- Introduction
- Stage Instructions
- Comparative Boolean Instructions
- Timer, Counter, and Shift Register Instructions

Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

## Introduction



This chapter provides information concerning the instructions used with RLLPLUS CPUs. If you are not familiar with RLLPLUS programming concepts, you should read Chapter 10 first. Chapter 10 will help you understand the basic concepts. The following table provides a quick reference listing of the instruction mnemonic and the page(s) defining the instruction. (The mnemonics are very similar to the instruction names and should be easy to become familiar with in a short time.) For example ISG is the mnemonic for Initial Stage. Each instruction definition will show in parentheses the keystrokes used to enter the instruction.

NOTE: Don't assume that the instructions in this chapter are the only ones you can use with your RLL PLUS CPU. There are many others that are discussed in Chapter 11 that you can use as well. If you are using a RLLPLUS CPU, such as the DL330P, then you should always consult this chapter before you use one of the instructions shown in Chapter 11. There may be differences in the way the instruction operates in an RLLPLUS CPU.

This chapter provides a description of several instructions that are similar, but slightly different from their RLL CPU counterparts. For example, you'll notice that a Counter instruction has two input lines in a RLL CPU but only one input line in an RLLPLUS CPU.

There are two ways to quickly find the instruction you need.

- If you know the instruction category (Stage, Comparative Boolean, etc.) just use the header at the top of the page to find the pages that discuss the instructions in that category.
- If you know the individual instruction mnemonic, use the following table to find the page that discusses the instruction.

| Instruction | Page |
| :--- | :--- |
| AND CNT | $12-17$ |
| AND SG | $12-9$ |
| AND TMR | $12-16$ |
| ANDN CNT | $12-17$ |
| ANDN SG | $12-9$ |
| ANDN TMR | $12-16$ |
| CNT | $12-19$ |
| ISG | $12-3$ |
| JMP | $12-5$ |
| NJMP | $12-5$ |
| OR CNT | $12-15$ |
| OR SG | $12-8$ |
| OR TMR | $12-14$ |
| ORN CNT | $12-15$ |
| ORN SG | $12-8$ |


| Instruction | Page |
| :--- | :--- |
| ORN TMR | $12-14$ |
| RST | $12-10$ |
| RST (counter) | $12-20$ |
| RST SG | $12-11$ |
| SET | $12-10$ |
| SET SG | $12-11$ |
| SG | $12-3$ |
| SR | $12-21$ |
| STR CNT | $12-13$ |
| STR TMR | $12-12$ |
| STR SG | $12-7$ |
| STRN CNT | $12-13$ |
| STRN SG | $12-7$ |
| STRN TMR | $12-12$ |
| TMR | $12-18$ |

## Stage Instructions

Initial Stage (ISG)
DL330P Only

Stage
(SG)
DL330P Only

The Initial Stage instruction is normally used as the first segment of a RLLPLUS program. Initial stages are activated when the CPU enters the run mode, this creates a starting point in the program. The Initial Stage can be made inactive by either jumping from it or resetting it. Multiple Initial Stages are allowed in a program.

The Stage instruction creates segments of a RLLPLUS program. Stages are activated by transitional logic, a jump or set stage executed from an active stage. Stages are de-activated one scan after transitional logic, a jump, or a reset stage instruction is executed.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :---: | :---: | :---: | :---: | :---: |
|  |  | aaaa | aaaa | aaaa |
| Stages | SG | -- | -- | $0-177$ |

RLL PLUS Instruction Set
Stage Instructions

The following example is a simple RLLPLUS program. This program utilizes the Initial Stage, Stage, and Jump instructions to create a structured program.


Handheld Programmer Keystrokes

| ISG | SHF | 0 | ENT |  |
| :---: | :---: | :---: | :---: | :---: |
| STR | SHF | 0 | ENT |  |
| OUT | SHF | 1 | 0 | ENT |
| STR | SHF | 1 | ENT |  |
| SET | SG | SHF | 2 | ENT |
| STR | SHF | 5 | ENT |  |
| JMP | SG | 1 | ENT |  |
| SG | SHF | 1 | ENT |  |
| STR | SHF | 2 | ENT |  |
| OUT | SHF | 1 | 1 | ENT |
| SG | SHF | 2 | ENT |  |
| STR | SHF | 6 | ENT |  |
| OUT | SHF | 1 | 2 | ENT |
| STR | SHF | 7 | ENT |  |
| AND | SG | 1 | ENT |  |
| JMP | SG | 0 | ENT |  |

Jump
(JMP)
DL330P Only

Not Jump
(NOT JMP) DL330P Only

The Jump instruction allows the program to transition from an active stage which contains the jump instruction to another which is specified in the instruction. The jump will occur when the input logic is true. The active stage that contains the Jump will be de-activated 1 scan after the Jump instruction is executed.

The Not Jump instruction allows the program to transition from an active stage which contains the jump instruction to another which is specified in the instruction. The jump will occur when the input logic is false. The active stage that contains the Not Jump will be de-activated 1 scan after the Not Jump instruction is executed.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :---: | :---: | :---: | :---: |
|  | aaaa | aaaa | aaaa |
| Stages | -- | -- | $0-177$ |

The following example is a simple RLL ${ }^{P L U S}$ program. This program utilizes the Initial Stage, Stage, Jump, and Not Jump instructions to create a structured program.


Handheld Programmer Keystrokes

| ISG | SG | SHF | 0 | ENT |
| :---: | :---: | :---: | :---: | :---: |
| STR | SHF | 0 | ENT |  |
| OUT | SHF | 1 | 0 | ENT |
| STR | SHF | 1 | ENT |  |
| JMP | SG | SHF | 1 | ENT |
| JMP | NOT | SG | SHF | 2 |
| SG | SHF | 1 | ENT |  |
| STR | SHF | 002 | ENT |  |
| OUT | SHF | 1 | 1 | ENT |
| STR | SHF | 3 | ENT |  |
| JMP | SG | SHF | 2 | ENT |
| SG | SHF | 2 | ENT |  |
| STR | SHF | 4 | ENT |  |
| OUT | SHF | 1 | 2 | ENT |
| STR | SHF | 5 | ENT |  |
| JMP | SG | SHF | 0 | ENT |

Store Stage (STR SG) DL330P Only

Store Not Stage (STR NOT SG) DL330P Only

The Store instruction begins a new rung or additional branch in a rung with a normally open stage contact. Status of the contact will be the same state as the associated Stage memory location.


The Store Not instruction begins a new rung or additional branch in a rung with a normally closed stage contact. Status of the contact will be opposite the state of the associated stage memory location.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :---: | :---: | :---: | :---: |
|  | aaaa | aaaa | aaaa |
| Stages | SG | -- | -- |

In the following Store example, when stage contact 000 is on, output 010 will energize.


In the following Store Not example, when stage contact 000 is off output 010 will energize.


Or Stage (OR SG) DL330P Only

Or Not Stage (OR NOT SG) DL330P Only

The Or instruction logically ors a normally open stage contact in parallel with another contact in a rung. The status of the contact will be the same state as the associated stage memory location.


The Or Not instruction logically ors a normally closed stage contact in parallel with another contact in a rung. The status of the contact will be opposite the state of the associated stage memory location.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |  |
| :---: | :---: | :---: | :---: | :---: |
|  |  | aaaa | aaaa | aaaa |
| Stages | SG | -- | -- | $0-177$ |

In the following Or example, when input 000 or stage contact 001 is on output 010 will energize.


Handheld Programmer Keystrokes


In the following Or Not example, when input 000 is on or stage contact 001 is off output 010 will energize.

Handheld Programmer Keystrokes

| STR | SHF | 0 | ENT |  |  |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| OR | NOT | SG | SHF | 1 | ENT |
| OUT | SHF | 1 | 0 | ENT |  |

And Stage (AND Stage) DL330P Only

And Not Stage (AND NOT SG) DL330P Only

The And instruction logically ands a normally open stage contact in series with another contact in a rung. The status of the contact will be the same state as the associated stage memory location.


The And Not instruction logically ands a normally closed stage contact in series with another contact in a rung. The status of the contact will be opposite the state of the associated stage memory location.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :---: | :---: | :---: | :---: |
|  |  | aaaa | aaaa |
| Stages | SG | - | - |

In the following And example, when input 000 and stage contact 001 is on output 010 will energize.


In the following And Not example, when input 000 is on and stage contact 001 is off output 010 will energize.


## Stage Instructions

## Set

(SET)
DL330P Only

The Set instruction sets or turns on a output or a consecutive range of outputs. Once the output is set it will remain on until it is reset using the Reset instruction. It is not necessary for the input controlling the Set instruction to remain on. The Set instruction is sometimes known as a latch. The Reset instruction is used to reset the output.

The Reset instruction resets or turns off an output or a consecutive range of outputs. Once the output is reset it is not necessary for the input to remain on. The Reset instruction is sometimes known as an unlatch instruction.


| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaaa | aaaa | aaaa |
| Outputs | -- | -- | $000-177$ |
|  |  | -- | $700-767$ |
| Control Relays | -- | $160-167$ |  |
|  |  |  | $170-174$ |
| $200-277$ |  |  |  |

In the following Set example, when input location 005 is on, outputs $20-37$ will be set on.


In the following Reset example, when input location 006 is on, outputs $020-37$ will be reset to the off state.


Set Stage (SET SG) DL330P Only

Reset Stage
(RST SG)
DL330P Only

The Set Stage instruction sets or turns on a stage or a consecutive range of stages. Once the stage is set it will remain on until a transition is made to another stage or the stage is reset using the Reset Stage instruction. It is not necessary for the input controlling the Set Stage instruction to remain on.

The Reset instruction resets or turns off a stage or a consecutive range of stages. Once the stage(s) is reset it is not necessary for the input to remain on.

| Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Stage | -- | -- | $000-177$ |

Optional memory range
SG aaa aaa


In the following Set Stage example, when input 000 is on, stages $30-47$ will be set on.


In the following Reset Stage example, when input 003 is on, stages $30-47$ will be reset off.


## Comparative Boolean Instructions

Store If Greater Than or Equal To Timer (STR TMR) DL330P Only

Store Not IF Greater Than Timer (STR NOT TMR) DL330P Only

The Store If Greater Than or Equal To instruction begins a new rung or additional branch in a rung with a normally open comparative timer contact. The contact will be on if the specified timer T aaa $\geq \mathrm{B}$ bbbb.

The Store Not If Greater Than instruction begins a new rung or additional branch in a rung with a normally closed comparative timer contact. The contact will be on if the specified timer T aaa $<\mathrm{B}$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :--- | :--- | :--- | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Timers | T | -- | -- | -- | -- | $600-677$ | -- |
| Data registers | R | -- | -- | -- | -- | $400-577$ |  |
| Constant | K | -- | -- | -- | -- | -- | $0-9999$ |

In the following Store If Greater Than or Equal To example, when $\mathrm{T} 602 \geq$ the value 1538 the contact will turn on and output 014 will energize.


Handheld Programmer Keystrokes


In the following Store Not If Greater Than example, when T602 < the value in R404 the contact will turn on and output 020 will energize.

Handheld Programmer Keystrokes

| STR | NOT | TMR | SHF | 6 | 0 | 4 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| R | 4 | 0 | 0 | 4 | ENT |  |
| RNT |  |  |  |  |  |  |
| OUT | SHF | 2 | 0 | ENT |  |  |

Store If Greater
Than or Equal
To Counter (STR CNT)
DL330P Only

The Store If Greater Than or Equal To instruction begins a new rung or additional branch in a rung with a normally open comparative counter contact. The contact will be on if the specified counter CT aaa $\geq \mathrm{B}$ bbbb.


The Store Not If Greater Than instruction begins a new rung or additional branch in a rung with a normally closed comparative counter contact. The contact will be on if the specified counter CT aaa $<\mathrm{B} \mathrm{bbbb}$.


In the following Store If Greater Than or Equal To example, when CT602 $\geq$ the value in R404 the contact will turn on and output 014 will energize.


In the following Store Not If Greater Than example, when CT602 < the constant value 4620 the contact will turn on and output 020 will energize.


Handheld Programmer Keystrokes

| STR | NOT | CNT | SHF | 6 | 0 | 2 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| SHF | 4 | 6 |  | 2 |  | 0 |

OUT SHF 2,0 ENT

Or If Greater Than or Equal To Timer (OR TMR) DL330P Only

Or Not If Greater Than Timer (OR NOT TMR) DL330P Only

The Or If Greater Than or Equal To instruction connects a normally open comparative timer contact in parallel with another contact. The contact will be on if the specified timer $T$ aaa $\geq \mathrm{Bbbb}$.


The Or Not If Greater Than instruction connects a normally closed comparative timer contact in parallel with another contact. The contact will be on if the specified timer $T$ aaa $<B$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Timers | T | -- | -- | -- | -- | $600-677$ | -- |
| Data registers | R | -- | -- | -- | -- | $400-577$ |  |
| Constant | K | -- | -- | -- | -- | $0-9999$ |  |

In the following Or If Greater Than or Equal To example, when input contact 001 is on or $\mathrm{T} 602 \geq$ the value 1234 the contact will turn on and output 014 will energize.


In the following Or Not If Greater Than example, when input contact 003 is on or T602 < the value in R404 the contact will turn on and output 020 will energize.


Handheld Programmer Keystrokes


Or If Greater Than or Equal
To Counter (OR CNT) DL330P Only

Or Not If Greater Than Counter (OR NOT CNT) DL330P Only

The Or If Greater Than or Equal To instruction connects a normally open comparative counter contact in parallel with another contact. The contact will be on if the specified counter CT aaa $\geq \mathrm{B}$ bbbb.


The Or Not If Greater Than instruction connects a normally closed comparative counter contact in parallel with another contact. The contact will be on if the specified counter CT aaa $<\mathrm{B}$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | CT | -- | -- | -- | -- | $600-677$ | -- |
| Data registers | R | -- | -- | -- | -- | $400-577$ |  |
| Constant | K | -- | -- | -- | -- | $0-9999$ |  |

In the following Or If Greater Than or Equal To example, when input contact 007 is on or CT602 $\geq$ the value in R404 the contact will turn on and output 014 will energize.


In the following Or Not If Greater Than example, when input contact 003 is on or CT602 < the constant value 4620 the contact will turn on and output 020 will energize.



And If Greater
Than or Equal
To Timer
(AND TMR)
DL330P Only

And Not If Greater Than Timer (AND NOT TMR) DL330P Only

The And If Greater Than or Equal To instruction connects a normally open comparative timer contact in series with another contact. The contact will be on if the specified timer $T$ aaa $\geq \mathrm{Bbbbb}$.

The And Not If Greater Than instruction connects a normally closed comparative timer contact in series with another contact. The contact will be on if the specified timer $T$ aaa $<B$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :--- | :--- | :--- | :--- | :--- | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Timers | T | -- | -- | -- | -- | $600-677$ | -- |
| Data registers | R | -- | -- | -- | -- | $400-577$ |  |
| Constant | K | -- | -- | -- | -- | -- | $0-9999$ |

In the following And If Greater Than or Equal To example, when input contact 001 is on and $\mathrm{T} 602 \geq$ the value 1234 the contact will turn on and output 014 will energize.


In the following Store Not If Greater Than example, when input contact 003 is on and T602 < the value in R404 the contact will turn on and output 020 will energize.

DirectSOFT Display


Handheld Programmer Keystrokes


And If Greater
Than or Equal
To Counter (AND CNT) DL330P Only

The And If Greater Than or Equal To instruction connects a normally open comparative counter contact in series with another contact. The contact will be on if the specified counter CT aaa $\geq \mathrm{B}$ bbbb.


The And Not If Greater Than instruction connects a normally closed comparative counter contact in series with another contact. The contact will be on if the specified counter CT aaa $<\mathrm{B}$ bbbb.


| Operand Data Type |  | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | B | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | CT | -- | -- | -- | -- | $600-677$ | -- |
| Data registers | R | -- | -- | -- | -- | $400-577$ |  |
| Constant | K | -- | -- | -- | -- | $0-9999$ |  |

In the following Or If Greater Than or Equal To example, when input contact 007 is on and CT602 $\geq$ the value in R404 the contact will turn on and output 014 will energize.


In the following Or Not If Greater Than example, when input contact 003 is on and CT602 < the constant value 4620 the contact will turn on and output 020 energize.


Handheld Programmer Keystrokes


## Timer, Counter, and Shift Register Instructions

Timer<br>(TMR)<br>DL330P Only

The Timer instruction used in the DL330P CPU provides a single input timer with a 0.1 second increment (0-999.9 seconds) in the normal operating mode, or a 0.01 second increment ( $0-99.99$ seconds) in the fast timer mode with relay 770 on. The timer will time up to the maximum value (999.9 or 99.99) as long as the input logic remains on, once the input logic turns off the timer will reset to 0 . There is no timer bit associated with this timer. Comparative boolean instructions must be used to monitor the current value of this timer.

| Operand Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Time | -- | -- | $600-677$ |

In the following Timer example when input contact 000 is on timer 600 will time up. When input contact 000 goes off the timer will reset to zero. The comparative instruction will monitor the current value of the timer and energize when the current value of the timer is greater than or equal to the constant K30.


Counter
DL330P Only

The Counter instruction used in the DL330P CPU provides a single input counter with a counting range of 0-9999. The counter will count up to 9999 and stop. The Reset Counter instruction must be used to reset this counter. There is no
 counter bit associated with this counter, so comparative boolean instructions must be used to monitor the current value of this counter.

| Operand Data Type | D3-330 Range | D3-340 Range | D3-330P Range |
| :--- | :---: | :---: | :---: |
|  | aaa | aaa | aaa |
| Counter | -- | -- | $600-677$ |

In the following Counter example when input contact 000 transitions from off to on counter 600 will increment by one. When input contact 001 is on the Reset Counter instruction will reset the counter to 0 . The comparative instruction will monitor the current value of the counter and energize when the current value of the counter $\geq$ the constant K2.


Reset Counter (RST)
DL330P Only

The Reset Counter instruction used in the DL330P CPU provides a reset for the counter instruction. One counter or a range of counters can be reset.


| Operand Data Type | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: |
|  | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Counters | -- | -- | -- | -- | $600-677$ | $600-677$ |

In the following Reset Counter example when input contact 001 is on the Reset Counter instruction will reset counter 600.


Handheld Programmer Keystrokes


Shift Register (SR)
DL330P Only

The Shift Register instruction shifts data through a predefined number of control relays. There are 77 control relays which can be used for internal control relays or shift register bits. There is no limit to the number of shift registers which can be used in a program, however the total number of bits used cannot exceed 77.
The Shift Register has three input contacts.

- Data - determines the value (1 or 0 ) that will enter the register
- Clock - shifts the bits one position on each low to high transition
- Reset -resets the Shift Register to all zeros.


With each off to on transition of the clock input, the bits which make up the shift register block are shifted by one bit position and the status of the data input is placed into the starting bit position of the shift register. The direction of the shift depends on the entry in the From and To fields. From 160 to 167 would define a shift right block of eight bits to be shifted from bit left to right. From 167 to 160 would define a shift left block of eight bits, but would shift from right to left. The maximum size of the shift register block is limited to 77 bits. There is no minimum block size.

| Operand Data Type | D3-330 Range |  | D3-340 Range |  | D3-330P Range |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: |
|  | aaa | bbbb | aaa | bbbb | aaa | bbbb |
| Shift Register Bits | -- | -- | -- | -- | $160-174$ | $160-174$ |
|  |  |  |  | $200-277$ | $200-277$ |  |

In the following example, when the clock input transitions from low to high the value in the Data input is placed in the first bit position of the shift register and the successive successive bits are shifted to the right. When the Reset input transitions from low to high the entire shift register is set to zeros.


## Maintenance and Troubleshooting

In This Chapter. . . .

- Maintenance
- CPU Indicators
- Power Indicator
- RUN Indicator
- CPU Indicator
- BATT Indicator
- Expansion Base Power
- Testing Output Points
— I/O Module Troubleshooting
— Noise Troubleshooting
- Machine Startup and Program Troubleshooting


## Maintenance

Air Quality Maintenance

CPU Battery Replacement



DL330, DL330P, DL340 CPU Battery Replacement

The DL305 is a low maintenance system requiring only a few periodic checks to ensure your system stays up and running without problems. There are two things you should check periodically.

- Air quality (cabinet temperature, etc.)
- CPU battery

The quality of the air your system is exposed to can affect system performance. If you have placed your system in an enclosure, check to se the ambient temperature is not exceeding the operating specifications. If there are filters in the enclosure, you should clean or replace them as necessary. A good guideline is to check your system environment every one to two months and make sure the environment meets the system operating specifications.

The CPU battery is used to retain the application program, data register, and retentive memory types. The life expectancy of this battery is five years.

NOTE: Before replacing your CPU battery, you should back-up your application program. This can be done by saving the program to hard/floppy disk on a personal computer or using the handheld programmer along with a cassette tape recorder. The CPU has a built-in capacitor to retain the memory for several minutes while the battery is being replaced. Saving the program prior to replacing the battery is just an added precaution.

WARNING: If the battery connector is not connected to the PC board or the battery is left out of the system, the indicator light will not notify you of the error. Be sure the battery is in place and the connector is firmly seated before you place the CPU back into the base.

To replace the CPU battery:

1. Turn power off to the system.
2. Wait 60 seconds then remove the CPU. Do not short any connectors or components on the CPU since it may alter the program memory.
3. Unlatch and tilt the clip covering the battery.
4. Pull the two wire battery connector from the PC board.
5. Remove the battery.

WARNING: Do not attempt to recharge the battery or dispose of it by fire. The battery may explode or release hazardous materials.

To install the CPU battery:

1. Plug the (keyed) two wire battery connector on the battery into the connector on the PC board.
2. Push gently till the connector snaps closed
3. Slide the battery under the battery retaining clip till the battery is positioned in the socket.
4. Push the retaining clip down over the battery snapping the clip over the edge of the PC board.
5. Note the date the battery was changed.

Battery Removal


## CPU Indicators

The DL305 CPUs have indicators on the front to help you diagnose problems with the system. The table below gives a quick reference of potential problems with each status indicator. Following the table will be a detailed analysis of each of these indicator problems.

| Indicator Status | Potential Problems |
| :--- | :--- |
| Power (off) | 1. Improper wiring <br> 2. Power supply fuse is blown <br> 3. Power supply/CPU is faulty <br> 4. Other component such as an I/O module has power <br> supply shorted <br> 5. Power budget exceeded for the base power supply |
| RUN <br> (will not come on) | 1. CPU programming error <br> 2. Key switch on handheld not in RUN mode |
| CPU (on) | 1. CPU defective <br> 2. Severe electrical noise interference |
| BATT (on) | CPU battery low |
| BATT (flashing) | CPU in update mode |

## Power Indicator

In general there are four reasons for the CPU power status LED to be OFF:

1. Power to the base is incorrect or is not applied.
2. The power supply has a blown fuse.
3. Base power supply is faulty.
4. Other component(s) have the power supply shut down. This problem could be in the base or in the I/O modules.
5. Power budget for the base has been exceeded.

## Incorrect Base Power

 Blown FuseIf the voltage to the power supply is not correct, the CPU may not operate properly or may not operate at all. If this is a new installation, first check the terminal strip on the local CPU base to insure the base is wired correctly. If it is wired for 110 VAC while using 220 VAC the power supply in the base will be damaged. If this has happened, you will need to replace your base. If the wiring is correctly installed for the AC or DC you are using, you should measure the voltage at the terminal strip to insure it is within specification for the base you are using. If the voltage is not correct shut down the system and correct the problem.

The fuse for base power is located behind the power supply cover at the right side of the base.

1. Remove power from the base.
2. Remove the two slotted insert screws from the front cover.
3. Remove and replace the 2 A 250 V fuse. (4A 250 V for the DC models)
4. Place the front cover back on the power supply and insert the screws.
5. Reapply power to the system.


Faulty Base Power Supply

Device or Module Causing the Power Supply to Shutdown

Power Budget Exceeded

CPU Firmware Update Mode

There is not a good check to test for a faulty power supply other than substituting a known good base to see if this corrects the problem. If you have experienced major power surges, it is possible the base and power supply have been damaged. If you suspect this is the cause of the power supply damage, a line conditioner which removes damaging voltage spikes should be used in the future.

It is possible a faulty module or external device using the system power can shut down the power supply.

To test for a device causing this problem:

- Turn off power to the base.
- Disconnect all external devices (example Data Communication Unit, Prom Writer Unit) from the CPU.
- Reapply power to the base.

If the Power LED does not operate normally the problem is most likely in one of the modules in the base. To isolate which module is causing the problem remove one module at a time till the Power LED operates normally. Follow the procedure below:

- Turn off power to the base.
- Remove a module from the base.
- Reapply power to the base.

This normally would not be the problem if the machine had been operating correctly for a considerable amount of time prior to the indicator going off. Power budgeting problems usually occur during system start-up when the PLC is under operation and the inputs/outputs are requiring more current than the base power supply can provide.

WARNING: The PLC may reset if the power budget is exceeded. If there is any doubt about the system power budget, please check it at this time. Exceeding the power budget can cause unpredictable results which can cause damage and injury. Verify the modules in the base operate within the power budget for the chosen base. You can find additional information on power budget calculations by reviewing Chapter 4.

The D3-330 CPU firmware can be updated by replacing the EPROM with the new firmware programmed into it. The D3-340 CPU firmware can be updated with a firmware update tool. When this tool is used, the BATT indicator will flash as the CPU is being updated. Contact AutomationDirect Technical support team at 770-844-4200 for assistance.

## RUN Indicator

If the CPU will not enter the run mode (the RUN indicator is off), the problem is usually in the application program unless the CPU has a fatal error in which case the CPU LED should be on.
Both of the programming devices, handheld programmer and PC programming package, will return a error message and depending on the error may also recommend an AUX function to run that will aid in further diagnosing the problem. A complete list of error codes can be found in Appendix B.

## CPU Indicator

If the CPU indicator is on, a fatal error has occurred in the CPU. Generally, this is not a programming problem but an actual hardware failure. You can power cycle the system to clear the error. If the error clears the system should be closely monitored and every effort should be made to try to determine the cause of the problem. You will find this problem is sometimes caused by high frequency electrical noise introduced into the CPU from a outside source. Check your system grounding and install electrical noise filters if the grounding is suspected. If power cycling the system does not reset the error or if the problem returns replace the CPU. The CPU indicator lights when the watchdog timer is not processed within 100 ms . The RUN output from the power supply will also turn off.

## BATT Indicator

If the BATT indicator is on, the CPU battery is low and needs to be replaced. The battery voltage is continuously monitored while the system voltage is being supplied. The detection circuit will be activated when the voltage drops to 2.5 volts and CPU operation will still continue as normal. Internal relay 377 energizes when the BATT indicator is on.
Procedures for how to replace the battery can be found earlier in this chapter.

## Expansion Base Power

Because a expansion base contains no CPU the only method of determining if the base power supply if functioning correctly is the run relay provided for that base. This relay can be connected to an input point on the local CPU base or an external warning indicator to monitor the expansion base power supply. If the power supply fails, the run relay will open. The procedures for troubleshooting the expansion bases are the same as a local CPU base. Refer to the Power Indicator section for procedures.

## Testing Output Points

Testing Output
Points

Output points can be set on or off in the DL305 series CPUs but they cannot be forced in such a way which overrides ladder logic. If you want to do an I/O check out independent of the application program follow the procedure below.

| Step | Action |
| :---: | :--- |
| 1 | If you are using the handheld programmer, change the keyswitch to <br> PRG, if you are using DirectSOFT select program mode. |
| 2 | Go to address 0 (handheld SHF NXT keys) . |
| 3 | Insert a "END" (handheld CLR SHF INS NXT keys) statement at ad- <br> dress 0. (This will cause program execution to occur only at address 0 <br> and prevent the application program from turning the I/O points on or <br> off). |
| 4 | Change to Run mode using the handheld programmer or DirectSOFT. |
| 5 | Use the programming device to set (turn on) or reset (turn off) the points <br> you wish to test. |
| 6 | When you finish testing I/O points go to address 0 (handheld SHF, NXT, <br> NXT keys) and delete the "END" statement (handheld keys DEL PRV) |

The following diagram shows the Handheld Programmer keystrokes used to test an output point.

WARNING: Depending on your application, forcing I/O points may cause unpredictable machine operation that can result in a risk of personal injury or equipment damage. Make sure you have taken all appropriate safety precautions prior to testing any I/O points.



Insert a END statement at the beginning of the Program. This disables the remainder of the program.

To monitor the output point on the handheld programmer use the following keystrokes


To turn on the output point use the following keystrokes


When the MON command is used, the LED display shows 16 consecutive status points. The MON command has designated this LED to be output number 20.
(20)


To turn off the output point use the following keystrokes

(20)


## I/O Module Troubleshooting

Important Notes About I/O Module Diagnostics

When troubleshooting the DL series I/O modules there are a few facts you should be aware of. These facts may assist you in quickly correcting an I/O problem.

- The output modules cannot detect shorted or open output points. If you suspect one or more points on a output module to be faulty, you should measure the voltage drop from the common to the suspect point.
Remember when using a Digital Volt Meter, leakage current from an output device such as a triac or a transistor must be considered. A point which is off may appear to be on if no load is connected the point.
- If the I/O status indicators on the modules are logic side indicators. This means the LED which indicates the on or off status reflects the status of the point in respect to the CPU. On a output module the status indicators could be operating normally while the actual output device (transistor, triac etc.) could be damaged. With an input module if the indicator LED is on, the input circuitry should be operating properly. To verify proper functionality check to see the LED goes off when the input signal is removed.
- Leakage current can be a problem when connecting field devices to I/O modules. False input signals can be generated when the leakage current of an output device is great enough to turn on the connected input device. To correct this install a resistor in parallel with the input or output of the circuit. The value of this resistor will depend on the amount of leakage current and the voltage applied but usually a 10 K to 20 K ohm resistor will work. Insure the wattage rating of the resistor is correct for your application.
- The easiest method to determine if a module has failed is to replace it if you have a spare. However, if you suspect another device to have caused the failure in the module, that device may cause the same failure in the replacement module as well. As a point of caution, you may want to check devices or power supplies connected to the failed module before replacing it with a spare module.


## Noise Troubleshooting

Electrical Noise Problems

Reducing
Electrical Noise
Noise is one of the most difficult problems to diagnose. Electrical noise can enter a system in many different ways and they fall into two categories, conducted or radiated. It may be difficult to determine how the noise is entering the system but the corrective actions for either of the types of noise problems are similar.

- Conducted noise is when the electrical interference is introduced into the system by way of a attached wire, panel connection ,etc. It may enter through an I/O module, a power supply connection, the communication ground connection, or the chassis ground connection.
- Radiated noise is when the electrical interference is introduced into the system without a direct electrical connection, much in the same manner as radio waves.

While electrical noise cannot be eliminated it can be reduced to a level that will not affect the system.

- Most noise problems result from improper grounding of the system. A good earth ground can be the single most effective way to correct noise problems. If a ground is not available, install a ground rod as close to the system as possible. Insure all ground wires are single point grounds and are not daisy chained from one device to another. Ground metal enclosures around the system. A loose wire is no more than a large antenna waiting to introduce noise into the system; therefore, you should tighten all connections in your system. Loose ground wires are more susceptible to noise than the other wires in your system. Review Chapter 2 Installation and Safety Guidelines if you have questions regarding how to ground your system.
- Electrical noise can enter the system through the power source for the CPU and I/O. Installing a isolation transformer for all AC sources can correct this problem. DC sources should be well grounded good quality supplies. Switching DC power supplies commonly generates more noise than linear supplies do.
- Separate input wiring from output wiring. Never run I/O wiring close to high voltage wiring.


## Machine Startup and Program Troubleshooting

Even after our your best attempts at creating application programs, there are still times when you need some assistance. This is especially true during machine startup and program troubleshooting. With the DL305 CPUs there are a few things that help make this task easier.

- Program Syntax Check-find problems before startup
- Pause Relay - monitor output status without enabling the actual output points or field devices
- End Statement - move the End statement to disable parts of the program.

Syntax Check Even though the Handheld Programmer and DirectSOFT provide error checking during program entry, you may want to check a program that has been modified. Both programming devices offer a way to check the program syntax. For example, you can use check the program syntax from a Handheld Programmer, or you can use the PLC Diagnostics menu option within DirectSOFT. This check will find a wide variety of programming errors. The following example shows how to use the syntax check with a Handheld Programmer.


## Execute the syntax check



Press CLR to display the address where the error occurred


Correct the problem and continue running the Syntax check until the E07 message no longer appears.

Using the Pause Relay

Special Relay 376 provides a quick way to allow the inputs (or other logic) to operate while disabling any output points used with an OUT instruction. The output image register is still updated, but the output status is not written to the modules. For example, you could make this conditional by adding an input contact or CR to control the instruction with a switch or a programming device. Or, you could just add the instruction without any conditions so the outputs would be disabled at all times.

Normal Program PAUSE disables 020 and 021


By using this relay, you can still monitor the output status with a programming device. The programming device will show that the output should be on, even though the CPU does not actually update the I/O point.

WARNING: This special relay only inhibits those outputs referenced by the OUT instruction. Output points referenced by the SET OUT instruction are not disabled.

END Instruction Placement

If you need a way to quickly disable part of the program, just insert an END statement prior to the portion that should be disabled. When the CPU encounters the END statement, it assumes that is the end of the program. The following diagram shows an example.

See Errata Sheet at the beginning of this

New END disables X10 and Y1
 file. A new sub-section has been added titled: "Reset the PLC to Factory Defaults".

## Quick Start Example

In This Appendix. . . .

- Step 1: Unpack the DL305 Equipment
- Step 2: Configure the 5-slot Base as the Local CPU Base
- Step 3: Install the CPU and I/O Modules
- Step 4: Wire the I/O Modules to the Field Devices
- Step 5: Remove the Terminal Strip Access Cover
— Step 6: Connect the Power Wiring
- Step 7: Connect the Handheld Programmer
- Step 8: Connect the Power Source
- Step 9: Enter the Example Program

Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

Now, you have the material necessary to become confident and productive with the DL305. The rest of this chapter is dedicated to laying out all of the pieces necessary to put together a complete system. It will highlight where specific chapters apply to questions you will typically have during your system configuration. This example is not intended to tell you everything you need to start-up your system, warnings and helpful tips are in the rest of the manual. It is only intended to give you a general picture of what you will need to do to get your system powered-up.

## Step 1: Unpack the DL305 Equipment

Unpack the DL305 equipment and verify you have the parts necessary to build your system. The minimum parts you will need are:
15 -slot base
1 Handheld programmer
1 CPU
1 D3-08ND2 discrete input module or D3-08SIM input simulator (If you use any other discrete input module it will be necessary for you to look up the wiring information for the module you are using.)
1 D3-08TD2 discrete output module (Any of the DL305 output modules can be used for this example since we will just be looking at the status indicators.)
1 Power cord (which you supply)


## Step 2: Configure the 5-slot Base as the Local CPU Base

The 5 slot base must be configured for base 1 (the base where the CPU resides). Identification of this base as the local CPU base is made by placing the base toggle switch in the 1,3 position as indicated below the toggle switch. Refer to Chapter 4 for more information on base switches.


## Step 3: Install the CPU and I/O Modules

Insert the CPU and I/O modules into the base as shown below. The CPU must go into the far right side of the base in the position next to the Power Supply.
When inserting components into the base, align the PC board(s) of the module with the grooves on the top and bottom of the base. Push the module straight into the base until it is firmly seated in the backplane connector.
Placement of 8 point discrete and relay modules are not critical and may go in any slot in the local CPU base. Limiting factors for other types of modules are discussed in Chapter 4. You must also make sure you do not exceed the power budget for each base in your system configuration. Power budgeting is also discussed in Chapter 4.


## Step 4: Wire the I/O Modules to the Field Devices

This step is not necessary if you are using an input simulator module. The toggle switch provides an external control point where you can interact with your system. Wire the I/O module to the field device prior to applying power to the system. (This will ensure that a point is not accidentally turned on during the wiring operation.) Wire the discrete input module as shown below. If you are using a module other than the D3-08ND2 you will need to refer to Chapter 6, Discrete Input Modules, for wiring information. Chapter 2, Installation and Safety Guidelines provides a list of I/O wiring guidelines. In the example below there is a discrete input module and a discrete output module in the base. The discrete input module is connected to an external switch.


## Step 5: Remove the Terminal Strip Access Cover

Remove the base terminal strip cover.


## Step 6: Connect the Power Wiring

If you are using 110VAC to power the base you must wire to the top 2 designated terminals. If you are using a 220VAC power source then you wire to the top and third designated terminals. You can find a detailed explanation of the terminal block on the bases in Chapter 2, Installation and Safety Guidelines.


WARNING:To minimize the risk of electrical shock, make sure the power source is disconnected before you connect the power wiring. Also, make sure you connect the power wiring correctly. The unit will be damaged if you connect 220 VAC to the 115 VAC terminals.

Wire the appropriate power connections to the base as shown below. Once wiring is completed insert the base terminal strip cover.


## Step 7: Connect the Handheld Programmer

Put the handheld programmer's key switch in the PRG position. Attach the handheld programmer directly to the front of the CPU making sure the port on the back of the programmer aligns properly with the port on the CPU and the programmer's latches connect with the slots in the base power supply. Apply power to the base. LEDs on the programmer will display indicating a good connection.


## Step 8: Connect the Power Source

Apply power to the system and ensure the CPU PWR indicator is on. If the indicator is not on, disconnect the system power and check the wiring connections. If the wiring connections are correct, refer to Chapter 13 for additional assistance.

WARNING: To minimize the risk of electrical shock, make sure the power source is disconnected before you check the power wiring.

## Step 9: Enter the Example Program

The switch wired to the input module and status indicator (LED) on the face of the output module will be the two I/O points used in the simple rung of ladder logic you will enter. The following diagram shows the ladder logic representation of the the program which will be entered on the handheld programmer.


Enter the key sequences on the handheld programmer as shown below.

| CLR | SHF | 3 | 4 | 8 | DEL | NXT | (Clears the CPU memory) <br> (Stores input 000) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| STR | SHF | 0 | ENT |  |  |  |  |
| OUT | SHF | 1 | 0 | ENT |  |  | (Outputs an on or off state to address 010) |

With the programmer's key switch in the PRG position, open and close the field input switch and observe that only the 0 LED on the input module turns on and off. This indicates the input signal is being received.
Now put the programmers key switch in the run position. The RUN LED on the programer's display will turn on. Open and close the field input switch and observe the 0 LED on the face of the input module and the 0 LED on the face of the output module both turn on and off. This indicates the program is accurately reflecting the signals which it is receiving from the field device.

## DL305 Error Codes

## In This Appendix. . . . <br> - Error Code Table

Handheld Programmer D3-HP \& D3-HPP have been retired as of 03/2021 \& 01/2018 respectively. Please consider Productivity, BRX, or CLICK series PLC systems as upgrades.

$\left.$| DL305 Error Code | Description |
| :--- | :--- |
| E01 <br> Invalid Keystrokes | Invalid keystroke or series of keystrokes entered into the handheld programmer. <br> Refer to the DL305 Handheld Programmer manual for assistance in the operation you <br> are trying to perform. |
| E02 <br> Input Point Programmed <br> as Output | An I/O point dedicated to an input module has been used as an output in the <br> application program. Change the I/O reference number in the program which is <br> causing the error. |
| E03 <br> Stack Overflow | The maximum number of instructions utilizing the internal stack has exceeded eight. <br> These instructions can be a combination of AND STRR, OR STRs and MCS/MCR <br> groups. Reduce the number of these instructions which are pushed onto the stack at <br> one time. |
| E05 (NON Stage) <br> Duplicate Coil <br> Reference | Two or more output coils have the same data type and number. Change the duplicate <br> coil to correct the error. Duplicate coil references are valid with the SET instruction. <br> E05 will be generated at the address of the second duplicated output. |
| E05 (Stage) <br> Duplicate Stage <br> Reference | Two or more Stages have the same reference number. Change the duplicate Stage <br> number to correct the error. E05 will be generated at the address of the second <br> duplicated stage number. |
| E06 <br> MCR/MCS Mismatch | The number of MCR instructions do not match the number of MCS instructions. Each <br> MCR must have an accompanying MCS. |
| E07 <br> Missing CNT or SR <br> Contact | A required input contact is missing from a CNT (example, RESET input) or a SR <br> instruction. |
| E08 <br> Invalid Data Values | The required data values for a TMR, CNT or SR are missing or incorrect. Refer to the <br> DL305 Programming Manual Set for details on these instructions. |
| E09 <br> Incomplete Program <br> Rung | The rung does not terminate with an output as required. Program an output to <br> terminate the rung properly. |
| E11 |  |
| Program Full | There is no available program addresses in memory. Reduce the size of the program. |
| E21 |  |
| Program Memory Parity |  |
| Error |  | | A parity error has occurred in the program memory of the CPU. Clear the memory |
| :--- |
| and reload the program. If the error reoccurs replace the CPU. Severe electrical noise |
| will cause this problem. | \right\rvert\, | The password stored in the CPU is invalid. Press the "CLR" key twice on the |
| :--- |
| handheld programmer and the password will be reset to 0000. Re-enter the password |
| if required. |

# Instruction Execution Times 

In This Appendix. . . .

- Introduction
- DL330 Instruction Execution Times
- DL330P Instruction Execution Times
- DL340 Instruction Execution Times


## Introduction

This appendix contains several tables that provide the instruction execution times for the DL330, DL330P, and DL340 CPUs. One thing you will notice is that many of the execution times depend on the type of data being used with the instruction. For example, some of the instructions have different execution times if you use a regular data register instead of a constant.
You'll also notice that some of the data instructions (such as DSTR) require differing amounts of execution time depending on the type of data. There are generally three options.

- Data Registers
- I/O Data Registers
- Constants

The following paragraphs may help you understand the differences between the register types.

## Data Registers

Some data registers are primarily used to hold variable data and are considered true data registers. For example, the registers that store the timer or counter current values, or just regular variable data would be considered as a data register. Don't think that you cannot load a bit pattern into these types of registers, you can. It's just that their primary use is as a data register. The following locations are considered as data registers.

| Type of Data | DL330 | DL330P | DL340 |
| :--- | :--- | :--- | :---: |
| Timer / Counter Current Values | R600 - R677 | R600 - R677 | R600 - R677 |
| User Data Words | R400 - R563 | R400 - R563 | R400 - R563 <br> R700 - R767 |

I/O Data Registers
You may recall that the I/O points are automatically mapped into data register locations. The following locations that contain this data are considered I/O registers and will take longer to execute with most instructions.

| Type of Data | DL330 | DL330P | DL340 |
| :--- | :--- | :--- | :--- |
| I/O Points | R000 - R016* | R000-R016* | R000-R017* |
| R070 - R 076 | R070-R076 | R070-R076 |  |
|  |  |  |  |

NOTE: $160-167$ can be used as I/O in a DL330 or DL330P CPU under certain conditions. 160-177 can be used as I/O in a DL340 CPU under certain conditions. You should consult Chapter 4 to determine which configurations allow the use of these points.

These points are normally used as control relays. You cannot use them as both control relays and as I/O points. Also, if you use these points as I/O, you cannot access these I/O points as a Data Register reference.

How to Read the Tables

Some of the instructions can have more than one parameter so the table shows execution times that depend on the amount and type of parameters. For example, the when you use the SET instruction to set a range of stages in a DL330P CPU, the execution time depends on how many stages are being set by the instruction.


| Instruction | Stage Instruction |  | Stage Instruction not activated by a Jump <br> instruction <br> (ex. power flow) |
| :--- | :--- | :--- | :--- |
| SET SG | $26.3+13.1 \mu \mathrm{~s} \times(\mathrm{n}-1)$ | $18.8 \mu \mathrm{~s}$ | Does not apply |
| RST SG | $26.3+13.1 \mu \mathrm{~s} \times(\mathrm{n}-1)$ | $18.8 \mu \mathrm{~s}$ | Does not apply |

Execution depends on numbers of locations and types of data used

## DL330 Instruction Execution Times

Basic Input Instructions

| Instruction | Execute | Disabled by MCR |
| :--- | :--- | :--- |
| STR | $6.6 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR NOT | $9.1 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND | $5.3 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND NOT | $8.4 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR | $6.6 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR NOT | $9.1 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR T/C | $10.3 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR NOT T/C | $12.8 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND T/C | $5.3 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND NOT T/C | $8.4 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR T/C | $6.6 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR NOT T/C | $9.1 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR (Comparative Contact) | $50.9 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR NOT (Comparative Contact) | $61.5 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND (Comparative Contact) | $59.1 \mu \mathrm{~s}$ | $6.2 \mu \mathrm{~s}$ |
| AND NOT (Comparative Contact) | $60.3 \mu \mathrm{~s}$ | $6.2 \mu \mathrm{~s}$ |
| OR (Comparative Contact) | $60.3 \mu \mathrm{~s}$ | $6.2 \mu \mathrm{~s}$ |
| OR NOT (Comparative Contact) | $62.5 \mu \mathrm{~s}$ | $6.2 \mu \mathrm{~s}$ |
| AND STR | $3.8 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR STR | $3.8 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| MCR | $5.0 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| MCS | $3.0 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |

Output Type Instructions

| Instruction | Execute | Not Executed |
| :--- | :--- | :--- |
| OUT | $7.5 \mu \mathrm{~s}$ | $7.5 \mu \mathrm{~s}$ |
| SET OUT | $10.0 \mu \mathrm{~s}$ | $10.0 \mu \mathrm{~s}$ |
| SET | $17.5 \mu \mathrm{~s}$ | $17.5 \mu \mathrm{~s}$ |
| RST | $9.3 \mu \mathrm{~s}$ | $9.3 \mu \mathrm{~s}$ |
| SET OUT RST | $19.3 \mu \mathrm{~s}$ | $19.3 \mu \mathrm{~s}$ |

Timer, Counters, and Shift Registers

| Instruction | Execute w/ <br> Constant | Execute w/ <br> Data Register | Execute w/ <br> I/O Register | Not <br> Executed |
| :--- | :--- | :--- | :--- | :--- |
| TMR | $90.9 \mu \mathrm{~s}$ | $458.8 \mu \mathrm{~s}$ | $700.0 \mu \mathrm{~s}$ | $27.1 \mu \mathrm{~s}$ |
| CNT | $92.9 \mu \mathrm{~s}$ | $465.6 \mu \mathrm{~s}$ | $706.8 \mu \mathrm{~s}$ | $27.1 \mu \mathrm{~s}$ |
| SR | $64.1 \mu \mathrm{~s}+16.6 \mu \mathrm{~s}$ times (\# of shifts) | $53.1 \mu \mathrm{~s}$ |  |  |

Data Operations

| Instruction |  | Execute w/ Data Register | Execute w/ I/O Register | Execute w/ Constant | Not Executed |
| :---: | :---: | :---: | :---: | :---: | :---: |
| DSTR <br> DSTR1 <br> DSTR2 <br> DSTR3 <br> DSTR5 | $\begin{aligned} & \hline \text { F50 } \\ & \text { F51 } \\ & \text { F52 } \\ & \text { F53 } \\ & \text { F55 } \end{aligned}$ | $80.7 \mu \mathrm{~s}$ <br> $63.8 \mu \mathrm{~s}$ <br> $95.0 \mu \mathrm{~s}$ <br> $96.6 \mu \mathrm{~s}$ <br> N/A | $321.9 \mu \mathrm{~s}$ $140.9 \mu \mathrm{~s}$ $172.2 \mu \mathrm{~s}$ $173.8 \mu \mathrm{~s}$ $326.2 \mu \mathrm{~s}$ | $\begin{aligned} & \hline 14.3 \mu \mathrm{~s} \\ & \mathrm{~N} / \mathrm{A} \\ & \mathrm{~N} / \mathrm{A} \\ & \mathrm{~N} / \mathrm{A} \\ & \mathrm{~N} / \mathrm{A} \end{aligned}$ | $\begin{aligned} & \hline 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| DOUT DOUT1 DOUT2 DOUT3 DOUT5 | $\begin{aligned} & \text { F60 } \\ & \text { F61 } \\ & \text { F62 } \\ & \text { F63 } \\ & \text { F65 } \end{aligned}$ | $52.6 \mu \mathrm{~s}$ <br> $39.1 \mu \mathrm{~s}$ <br> $39.8 \mu \mathrm{~s}$ <br> $55.0 \mu \mathrm{~s}$ <br> N/A | $329.4 \mu \mathrm{~s}$ $160.1 \mu \mathrm{~s}$ 116.0 us $108.1 \mu \mathrm{~s}$ $358.3 \mu \mathrm{~s}$ | N/A <br> N/A <br> N/A <br> N/A <br> N/A | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| $\begin{aligned} & \hline \mathrm{CMP}<=> \\ & \text { ADD } \\ & \text { SUB } \\ & \text { MUL } \\ & \text { DIV } \end{aligned}$ | $\begin{aligned} & \hline \text { F70 } \\ & \text { F71 } \\ & \text { F72 } \\ & \text { F73 } \\ & \text { F74 } \end{aligned}$ | $\begin{array}{\|l} \hline 112.8 \mu \mathrm{~s} \\ 456.8 \mu \mathrm{~s} \\ 315.8 \mu \mathrm{~s} \\ 290-2664 \mu \mathrm{~s} \\ 742-2645 \mu \mathrm{~s} \end{array}$ | $\begin{array}{\|l\|} \hline 354.0 \mu \mathrm{~s} \\ 698.0 \mu \mathrm{~s} \\ 557.0 \mu \mathrm{~s} \\ 497-2851 \mu \mathrm{~s} \\ 1218-2851 \mu \mathrm{~s} \end{array}$ | $\begin{array}{\|l\|} \hline 57.0 \mu \mathrm{~s} \\ 262.0 \mu \mathrm{~s} \\ 275.0 \mu \mathrm{~s} \\ 223-2576 \mu \mathrm{~s} \\ 720-2557 \mu \mathrm{~s} \end{array}$ | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| DAND DOR |  | $\begin{aligned} & 103.7 \mu \mathrm{~s} \\ & 103.7 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & 345.0 \mu \mathrm{~s} \\ & 345.0 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & 55.6 \mu \mathrm{~s} \\ & 55.6 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| $\begin{aligned} & \text { SHFR } \\ & \text { SHFL } \end{aligned}$ |  | $\begin{aligned} & 216 \mu \mathrm{~s}+13.4 \mu \mathrm{~s} \\ & 220 \mu \mathrm{~s}+13.4 \mu \mathrm{~s} \end{aligned}$ | ( \# of shifts) <br> ( \# of shifts) |  | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| $\begin{aligned} & \text { DECO } \\ & \text { ENCO } \end{aligned}$ | F82 | $\begin{array}{\|l\|} \hline 56.3 \mu \mathrm{~s} \\ 282.0 \mu \mathrm{~s} \end{array}$ | $\begin{array}{\|l} \hline \mathrm{N} / \mathrm{A} \\ \mathrm{~N} / \mathrm{A} \end{array}$ | $\begin{array}{\|l\|l\|} \hline \mathrm{N} / \mathrm{A} \\ \mathrm{~N} / \mathrm{A} \end{array}$ | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| INV <br> BIN <br> BCD | F84 F85 F86 | $\begin{aligned} & 30.0 \mu \mathrm{~s} \\ & 412.2 \mu \mathrm{~s} \\ & 746.0 \mu \mathrm{~s} \end{aligned}$ | N/A <br> N/A <br> N/A | N/A <br> N/A <br> N/A | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| FAULT | F20 | $114.0 \mu \mathrm{~s}$ | $355.3 \mu \mathrm{~s}$ | 72.2 us | $6.3 \mu \mathrm{~s}$ |

## DL330P Instruction Execution Times

Basic Input
Instructions

| Instruction | I/O, Control Relay |  | Stage |  | Timer / Counter |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | Executed | Not Executed | Executed | Not Executed | Executed | Not Executed |
|  | * ** | * ** | * ** | ** | * ** | ** |
| STR | 28.4 / 31.4 ${ }^{\text {us }}$ | 21.3 us | 25.6 / 30.9 ${ }^{\text {us }}$ | 22.2 us | 121.3/117.8 $\mu \mathrm{s}$ | 28.4 us |
| STR NOT | 28.4 / 31.4 4 s | 21.3 us | 25.6 / 30.9 ${ }^{\text {us }}$ | 22.2 us | 121.3/117.8us | 28.4 us |
| AND | 13.4 / 20.0us | 13.4 / 20.0 $\mu \mathrm{s}$ | 13.4 / 20.0us | 13.4/20.0us | 123.1/119.6 ${ }^{\text {s }}$ | 20.3 us |
| AND NOT | 18.1/21.6us | 10.3 us | 18.1 / 21.6us | 10.3 us | 123.1/119.6 ${ }^{\text {s }}$ | 20.3 us |
| OR | 21.8/25.3us | 14.7 us | 21.8/25.3 ${ }^{\text {us }}$ | 14.7 us | 123.1/119.6us | 20.3 us |
| OR NOT | 20.6 / 24.1 us | 14.7 us | 20.6 / 24.1 1 s | 14.7 us | 123.1/119.6 ${ }^{\text {s }}$ | 20.3 us |

* Execution time when data type is ON. For example, STR 000 takes $28.4 \mu \mathrm{~s}$ if point 000 is on.
** Execution time when data type is OFF. For example, STR 000 takes $31.4 \mu$ s if point 000 is off.

| Instruction | Executed | Not Executed |
| :--- | :--- | :--- |
| AND STR | $25.9 \mu \mathrm{~s}$ | $22.8 \mu \mathrm{~s}$ |
| OR STR | $25.9 \mu \mathrm{~s}$ | $22.8 \mu \mathrm{~s}$ |

Output Type Instructions

| Instruction | Execute | Not Executed |
| :--- | :--- | :--- |
| OUT | $20.6 \mu \mathrm{~s}$ | $20.6 \mu \mathrm{~s}$ |
| SET OUT | $24.3 \mu \mathrm{~s}$ | $16.6 \mu \mathrm{~s}$ |
| SET | $24.3 \mu \mathrm{~s}$ | $16.6 \mu \mathrm{~s}$ |
| RST | $24.3 \mu \mathrm{~s}$ | $16.6 \mu \mathrm{~s}$ |
| SET OUT RST | $33.8 \mu \mathrm{~s}$ | $29.4 \mu \mathrm{~s}$ |

Timer, Counters, and Shift Registers

| Instruction | Execute | Not Executed |
| :--- | :--- | :--- |
| TMR | $92.8 \mu \mathrm{~s}$ | $50.9 \mu \mathrm{~s}$ |
| CNT | $97.5 \mu \mathrm{~s}$ | $46.3 \mu \mathrm{~s}$ |
| RST CNT | $25.9 \mu \mathrm{~s}$ | $16.6 \mu \mathrm{~s}$ |
| SR | $75.9+11.5 \mu \mathrm{~s} \times$ (\# of shifts) | $41.9 \mu \mathrm{~s}$ |

Stage Instructions

| Instruction | Stage Instruction |  | Stage Instruction not activated by a Jump <br> instruction <br> (ex. power flow) |  |
| :--- | :--- | :--- | :--- | :--- |
|  | Executed | Not Executed | Executed | Not Executed |
| ISG | $35.3 \mu \mathrm{~s}$ | $20.0 \mu \mathrm{~s}$ | $50.9 \mu \mathrm{~s}$ | $30.6 \mu \mathrm{~s}$ |
| SG | $35.3 \mu \mathrm{~s}$ | $20.0 \mu \mathrm{~s}$ | $50.9 \mu \mathrm{~s}$ | $30.6 \mu \mathrm{~s}$ |
| JMP | $28.4 \mu \mathrm{~s}$ | $16.6 \mu \mathrm{~s}$ | Does not apply |  |
| NJMP | $40.3 \mu \mathrm{~s}$ | $28.4 \mu \mathrm{~s}$ | Does not apply |  |
| SET SG | $26.3+13.1 \mu \mathrm{~s} \times(\mathrm{n}-1)$ | $18.8 \mu \mathrm{~s}$ | Does not apply |  |
| RST SG | $26.3+13.1 \mu \mathrm{~s} \times(\mathrm{n}-1)$ | $18.8 \mu \mathrm{~s}$ | Does not apply |  |

Data Operation Instructions

| Instruction |  | Execute w/ Data Register | Execute w/ I/O Register | Execute w/ Constant | Not Executed |
| :---: | :---: | :---: | :---: | :---: | :---: |
| DSTR <br> DSTR1 <br> DSTR2 <br> DSTR3 <br> DSTR5 | $\begin{aligned} & \hline \hline \text { F50 } \\ & \text { F51 } \\ & \text { F52 } \\ & \text { F53 } \\ & \text { F55 } \end{aligned}$ | $80.7 \mu \mathrm{~s}$ <br> $63.8 \mu \mathrm{~s}$ <br> $95.0 \mu \mathrm{~s}$ <br> $96.6 \mu \mathrm{~s}$ <br> N/A | $321.9 \mu \mathrm{~s}$ 140.9 us 172.2 us 173.8 us $326.2 \mu \mathrm{~s}$ | $\begin{aligned} & \hline 14.3 \mu \mathrm{~s} \\ & \mathrm{~N} / \mathrm{A} \\ & \mathrm{~N} / \mathrm{A} \\ & \mathrm{~N} / \mathrm{A} \\ & \mathrm{~N} / \mathrm{A} \end{aligned}$ | $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ |
| DOUT <br> DOUT1 <br> DOUT2 <br> DOUT3 <br> DOUT5 | $\begin{aligned} & \hline \text { F60 } \\ & \text { F61 } \\ & \text { F62 } \\ & \text { F63 } \\ & \text { F65 } \end{aligned}$ | $52.6 \mu \mathrm{~s}$ <br> $39.1 \mu \mathrm{~s}$ <br> $39.8 \mu \mathrm{~s}$ <br> $55.0 \mu \mathrm{~s}$ <br> N/A | $\begin{aligned} & 329.4 \mu \mathrm{~s} \\ & 160.1 \mu \mathrm{~s} \\ & 116.0 \mu \mathrm{~s} \\ & 108.1 \mu \mathrm{~s} \\ & 358.3 \mu \mathrm{~s} \end{aligned}$ | N/A <br> N/A <br> N/A <br> N/A <br> N/A | 6.3 us <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ |
| $\begin{aligned} & \text { CMP<=> } \\ & \text { ADD } \\ & \text { SUB } \\ & \text { MUL } \\ & \text { DIV } \end{aligned}$ | $\begin{aligned} & \hline \text { F70 } \\ & \text { F71 } \\ & \text { F72 } \\ & \text { F73 } \\ & \text { F74 } \end{aligned}$ | $\begin{aligned} & \hline 112.8 \mu \mathrm{~s} \\ & 456.8 \mu \mathrm{~s} \\ & 315.8 \mu \mathrm{~s} \\ & 290-2664 \mu \mathrm{~s} \\ & 742-2645 \mathrm{~s} \end{aligned}$ | $\begin{aligned} & \hline 354.0 \mu \mathrm{~s} \\ & 698.0 \mu \mathrm{~s} \\ & 557.0 \mu \mathrm{~s} \\ & 497-2851 \mu \mathrm{~s} \\ & 1218-2851 \mu \mathrm{~s} \end{aligned}$ | $\begin{array}{\|l\|} \hline 57.0 \mu \mathrm{~s} \\ 262.0 \mu \mathrm{~s} \\ 275.0 \mu \mathrm{~s} \\ 223-2576 \mu \mathrm{~s} \\ 720-2557 \mu \mathrm{~s} \end{array}$ | $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ <br> $6.3 \mu \mathrm{~s}$ |
| DAND <br> DOR |  | $\begin{aligned} & 103.7 \mu \mathrm{~s} \\ & 103.7 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & 345.0 \mu \mathrm{~s} \\ & 345.0 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & 55.6 \mu \mathrm{~s} \\ & 55.6 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| $\begin{aligned} & \hline \text { SHFR } \\ & \text { SHFL } \end{aligned}$ | F80 | $216 \mu \mathrm{~s}+13.4 \mu \mathrm{~s}$ times (\# of shifts) $220 \mu \mathrm{~s}+13.4 \mu \mathrm{~s}$ times (\# of shifts) |  |  | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| $\begin{aligned} & \text { DECO } \\ & \text { ENCO } \end{aligned}$ | F82 | $\begin{aligned} & 56.3 \mu \mathrm{~s} \\ & 282.0 \mu \mathrm{~s} \end{aligned}$ | $\begin{aligned} & \hline N / A \\ & N / A \end{aligned}$ | $\begin{array}{\|l\|} \hline \mathrm{N} / \mathrm{A} \\ \mathrm{~N} / \mathrm{A} \end{array}$ | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| INV <br> BIN <br> BCD | $\begin{aligned} & \text { F84 } \\ & \text { F85 } \\ & \text { F86 } \end{aligned}$ | $\begin{aligned} & 30.0 \mu \mathrm{~s} \\ & 412.2 \mu \mathrm{~s} \\ & 746.0 \mu \mathrm{~s} \end{aligned}$ | N/A <br> N/A <br> N/A | N/A <br> N/A <br> N/A | $\begin{aligned} & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \\ & 6.3 \mu \mathrm{~s} \end{aligned}$ |
| FAULT | F20 | $114.0 \mu \mathrm{~s}$ | $355.3 \mu \mathrm{~s}$ | 72.2 us | $6.3 \mu \mathrm{~s}$ |

## DL340 Instruction Execution Times

Basic Input Instructions

| Instruction | Execute | Disabled by MCR |
| :--- | :--- | :--- |
| STR | $0.875 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR NOT | $1.750 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND | $0.625 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND NOT | $1.5 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR | $1.125 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR NOT | $1.75 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR T/C | $0.875 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR NOT T/C | $1.75 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND T/C | $0.625 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND NOT T/C | $1.5 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR T/C | $1.125 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR NOT T/C | $1.75 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND STR | $0.75 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| OR STR | $0.75 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| MCR | $0.75 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| MCS | $1.125 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |

Comparative Contacts

| Instructions | Execute w/ <br> Data <br> Register | Execute w/ <br> I/O Register | Execute w/ <br> Constant |  | Not <br> Executed |
| :--- | :--- | :--- | :--- | :--- | :--- |
|  |  |  | EE $/$ UV |  |  |
| STR | $56.8 \mu \mathrm{~s}$ | $95.0 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| STR NOT | $56.8 \mu \mathrm{~s}$ | $96.5 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $\mathrm{~N} / \mathrm{A}$ |
| AND | $56.8 \mu \mathrm{~s}$ | $95.0 \mu \mathrm{~s}$ | $15.0 \mu \mathrm{~s}$ | $15.0 \mu \mathrm{~s}$ | $1.4 \mu \mathrm{~s}$ |
| AND NOT | $56.8 \mu \mathrm{~s}$ | $96.5 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $1.4 \mu \mathrm{~s}$ |
| OR | $56.8 \mu \mathrm{~s}$ | $94.0 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $15.6 \mu \mathrm{~s}$ | $1.4 \mu \mathrm{~s}$ |
| OR NOT | $56.8 \mu \mathrm{~s}$ | $94.0 \mu \mathrm{~s}$ | $16.2 \mu \mathrm{~s}$ | $16.2 \mu \mathrm{~s}$ | $1.4 \mu \mathrm{~s}$ |

Output Type Instructions

| Instruction | Execute | Not Executed |
| :--- | :--- | :--- |
| OUT | $1.188 \mu \mathrm{~s}$ | $1.188 \mu \mathrm{~s}$ |
| SET OUT | $1.563 \mu \mathrm{~s}$ | $1.563 \mu \mathrm{~s}$ |
| SET | $1.625 \mu \mathrm{~s}$ | $1.4 \mu \mathrm{~s}$ |
| RST | $1.625 \mu \mathrm{~s}$ | $1.4 \mu \mathrm{~s}$ |
| SET OUT RST | $7.5 \mu \mathrm{~s}$ | $7.125 \mu \mathrm{~s}$ |

Timer, Counters, and Shift Registers

| Instructions | Execute w/ <br> Data <br> Register | Execute w/ <br> I/O Register | Execute w/ <br> Constant |  | Not <br> Executed |
| :--- | :--- | :--- | :--- | :--- | :--- |
|  |  |  | EE / UV |  |  |
| TMR | $68.1 \mu \mathrm{~s}$ | $113.8 \mu \mathrm{~s}$ | $22.5 \mu \mathrm{~s}$ | $22.5 \mu \mathrm{~s}$ | $15.7 \mu \mathrm{~s}$ |
| CNT | $67.3 \mu \mathrm{~s}$ | $97.9 \mu \mathrm{~s}$ | $22.5 \mu \mathrm{~s}$ | $22.5 \mu \mathrm{~s}$ | $25.6 \mu \mathrm{~s}$ |
| SR | $21.8 \mu \mathrm{~s}+3.8 \mu \mathrm{~s}$ times ( \# of shifts) | $8.3 \mu \mathrm{~s}$ |  |  |  |

Data Operation Instructions


# DL305 Product Weight Tables 

In This Appendix. . . .
— Product Weight Table

## Product Weight Table

| CPUs | Weight |
| :---: | :---: |
| D3-330 | 6.3 oz. (178g) |
| D3-330P | 6.3 oz. (178g) |
| D3-340 | 5.2 oz. (146g) |
| Specialty CPUs |  |
| F3-OMUX-1 | 6.4 oz. (182g) |
| F3-OMUX-2 | 6.4 oz. (182g) |
| F3-PMUX | 3.7 oz. (104g) |
| F3-RTU | 6.7 oz. (190g) |
| Bases |  |
| D3-05B | 34.0 oz. (964g) |
| D3-05BDC | 34.0 oz. $(964 \mathrm{~g})$ |
| D3-08B | $\begin{aligned} & 44.2 \mathrm{oz} . \\ & (1253 \mathrm{~g}) \end{aligned}$ |
| D3-10B | $\begin{aligned} & 50.5 \mathrm{oz} . \\ & (1432 \mathrm{~g}) \end{aligned}$ |
| DC Input Modules |  |
| D3-08ND2 | 4.2 oz. (120g) |
| D3-16ND2-1 | 6.3 oz. (178g) |
| D3-16ND2-2 | 5.3 oz. (150g) |
| D3-16ND2F | 6.3 oz. (178g) |
| F3-16ND3F | 5.4 oz. (153g) |
| AC Input Modules |  |
| D3-08NA-1 | 5 oz. (140g) |
| D3-08NA-2 | 5 oz ( 140 g ) |
| D3-16NA | 6.4 oz. (180g) |
| AC/DC Input Modules |  |
| D3-08NE3 | $4.2 \mathrm{oz} .(120 \mathrm{~g})$ |
| D3-16NE3 | 6 oz. (170g) |


| DC Output Modules | Weight |
| :---: | :---: |
| D3-08TD1 | 4.2 oz. (120g) |
| D3-08TD2 | 4.2 oz. (120g) |
| D3-16TD1-1 | 5.6 oz. (160g) |
| D3-16TD1-2 | 5.6 oz. (160g) |
| D3-16TD2 | 7.1 oz. (210g) |
| AC Output Modules |  |
| D3-04TAS | 6.4 oz. (180g) |
| F3-08TAS | 6.3 oz. (178g) |
| F3-08TAS-1 | 6.3 oz. (178g) |
| D3-08TA-1 | 7.4 oz. (210g) |
| D3-08TA-2 | 6.4 oz. (180g) |
| F3-16TA-2 | 7.7 oz. (218g) |
| D3-16TA-2 | 7.2 oz. (210g) |
| Relay Output Modules |  |
| D3-08TR | 7 oz. (200g) |
| F3-08TRS-1 | 8.9 oz. (252g) |
| F3-08TRS-2 | 9 oz. (255g) |
| D3-16TR | 8.5 oz. (248g) |
| Analog Modules |  |
| D3-04AD | 7 oz. (200g) |
| F3-04ADS | 6.9 oz. (195g) |
| F3-08AD | 5.5 oz. (154g) |
| F3-08TEMP | 5.2 oz. (147g) |
| F3-08THM-n | 6 oz. (170g) |
| F3-16AD | 5.4 oz. (152g) |
| D3-02DA | 7 oz. (200g) |
| F3-04DA-1 | 6.3 oz. (180g) |
| F3-04DA-2 | 6.3 oz. (180g) |
| F3-04DAS | 7 oz. (200g) |


| Communications <br> and Networking | Weight |
| :--- | :--- |
| D3-232-DCU | 15.0 oz. $(427 \mathrm{~g})$ |
| D3-422-DCU | 14.8 oz. $(419 \mathrm{~g})$ |
| ASCII BASIC <br> Modules |  |
| F3-AB128-R | 5.1 oz. $(146 \mathrm{~g})$ |
| F3-AB128-T | 6.2 oz. $(175 \mathrm{~g})$ |
| F3-AB128 | 5.4 oz. $(154 \mathrm{~g})$ |
| Specialty <br> Modules | 3.0 oz. $(85 \mathrm{~g})$ |
| D3-08SIM | $5.2 \mathrm{oz} .(147 \mathrm{~g})$ |
| D3-HSC | $13.0 \mathrm{oz} .(368 \mathrm{~g})$ |
| D3-PWU | $10 \mathrm{oz} .(30 \mathrm{~g})$ |
| D3-FILL | 7. |
| Programming | 7.2 oz. $(204 \mathrm{~g})$ |
| D3-HP |  |
| D3-HPP |  |

## Index

## A

Accumulator
load and output instructions, 11-25
logic instructions, 11-30
operations, 9-10-9-13
shifting bits in, 11-42
Adding Numbers, 11-34
Agency Approvals, 2-7
Aliases, 8-25
Auxiliary Functions, 3-17

## B

## Bases

expansion, 4-14
I/O supported, 4-8
installation spacing, 2-4
installing modules, 2-10
local, 4-14
mounting dimensions, 2-10, 4-8
power budget, 4-26-4-31
power specifications, 2-6
power supply schematics, 4-11
power wiring, 2-11
run relay, 4-12
setting base jumpers, 4-16
setting switches, 4-16
specifications, 4-10-4-13
Battery, replacement, 3-14, 13-2
Baud Rate, 3-12
Bit Operation Instructions, 11-42
Boolean Instructions, 9-3, 11-4-11-18

## C

Communication, instructions, 11-52
Communication Port
data format, 3-13
DL340, 3-12
DL340 port diagrams, 3-13
master / slave selection, 3-13
response delay time, 3-13
Comparative Boolean Instructions, 11-19-11-21
in stages, 10-18, 12-12-12-19
Configuration, I/O examples, 4-17-4-25
Control Relays, 8-21
Converting Number Formats, 11-44-11-49
Counters, 8-22, 9-9, 11-23
in stages, 10-15, 12-19
CPU
auxiliary functions, 3-17
battery, 3-14
clearing memory, 3-21
features, 3-2
indicators, 13-3-13-6
memory options, 3-5
mode setting, 8-4
modes of operation, 3-20, 8-6-8-8
operating system, 8-3
scan time, 8-16
setup
DL330/DL340P, 3-9
DL340, 3-10
DL340 network address, 3-12
setup and system functions, 3-16
specifications, 3-3
switches
DL330/DL330P, 3-9
DL340, 3-10-3-13

## D

Data Instructions, 9-12-9-47
in stages, 10-17
Data Registers, 8-23
Derating Characteristics, 5-10
Dimensions, 2-7
Discrete Input
specifications, 6-4-6-15
terminology, 6-2
Discrete Memory, 8-19
Discrete Output
specifications, 7-6-7-20
terminology, 7-2
Dividing Numbers, 11-40

## E

EEPROM, 3-5
Enclosures, selection, 2-7
End Instruction, 9-3
Environmental Specifications, 2-6
Error Codes, B-2
Execution Times, 8-18, C-2-C-9
Expansion Bases, 4-14-4-15

## F

Fault Messages, 11-56
Flowchart Programming, 10-26
Forcing I/O, 8-10
Fuses
I/O protection, 5-12
power supply, 13-4

## G

Grounding, 2-4

I/O Memory, 8-20
I/O Modules
address switch (base), 4-16
configuration history, 4-2
derating, 5-10
discrete input specifications, 6-4-6-15
discrete output specifications, 7-6-7-20
example configurations, 4-17-4-25
fuse protection, 5-12-5-15
installing, 4-13
numbering, 4-2, 4-3
placement, 4-4-4-6
point requirements, 4-3
power requirements, 4-27-4-29
response time, 8-14
selection considerations, 5-2
sinking and sourcing circuits, 5-2
solid state field devices, 5-9
testing outputs, 13-8
troubleshooting, 13-10
update sequence, 8-8
wiring guidelines, 2-12, 5-11
Indicators, CPU, 13-3-13-6
Input Modules
specifications, 6-4-6-15
wiring diagrams, 6-4-6-15
Installation
base mounting dimensions, 2-10
base power wiring, 2-11
base wiring, 4-7-4-9
component dimensions, 2-7
DL330/DL330P setup, 3-9
DL340 setup, 3-10
grounding, 2-4
I/O modules, 4-13
I/O wiring guidelines, 2-12
installing modules, 2-10
local and expansion bases, 4-14
panel design specifications, 2-4
setting CPU switches, 3-9-3-12
Instruction, execution times, 8-18, C-2-C-9
Instruction Set, index, 11-3
Instructions
accumulator load and output, 11-25
accumulator logic instructions, 11-30
bit operations, 11-42
boolean, 11-4-11-18
comparative boolean, 11-19-11-21
in stages, 12-12-12-19
counters, 11-23
in stages, 12-19
end placement, 9-3, 13-13
initial stage, 12-3
jump, 12-5
math, 11-34
messages (fault), 11-56
network communication, 11-52
not jump, 12-5
number conversion, 11-44
program control, 11-50
RLLPLUS, 12-2
shift registers, 11-24
in stages, 12-21
stage, 12-3
timers, 11-22
in stages, 12-18

## J

Jump Instruction, 12-5
Jumpers, on bases, 4-16

## L

Latching Outputs, in stages, 10-14
Local Bases, 4-14

## M

Maintenance
battery replacement, 13-2
guidelines, 13-2
Master Control Relays, 11-50
Math Instructions, 11-34
Memory
battery backup, 3-14
clearing, 3-21
external storage, 3-4-3-6
initialization, 8-5
maps, 8-19, 8-25-8-40
options for CPUs, 3-4-3-6
PROM Writer Unit, 3-6
retentive, 3-4, 3-11, 8-5
retentive selection switch, 3-9
volatile and non-volatile, 3-4
Messages, 11-56
Multiplying Numbers, 11-38

## N

Network Address, 3-12, 3-13
Network Instructions, 11-52
Noise, reducing problems, 13-11
Not Jump Instruction, 12-5
Number Conversion Instructions, 11-44


Output Modules
specifications, 7-6-7-20
using outputs in stages, 10-12
wiring diagrams, 7-6-7-20

## P

Pause Relay, 13-13
Power Budget, 4-26-4-31 worksheet, 4-31
Power Specifications, 2-6
Power Supply
schematics, 4-11
wiring, 4-9-4-12
Program Control Instructions, 11-50
Program Mode, 8-6
Programming
accumulator usage, 9-10-9-12
basic concepts, 9-2
counters, 9-9
device connections, 3-18
flowchart style, 10-26
instruction set index, 11-3
RLLPLUS concepts, 10-2-10-37
stack operation, 9-6
timers, 9-8
troubleshooting, 13-12
PROM Writer Unit, 3-6

Quick Start, A-2-A-7

## R

RAM, 3-5
Retentive Memory, 3-4, 3-11
initialization of, 8-5
selection switch, 3-9
Run Mode, 8-7
Run Relay, 4-12

## S

## Safety

fuses, 5-12
guidelines, 2-2
levels of protection, 2-2
panel design specifications, 2-4
planning for, 2-2
sources of assistance, 2-2
Scan Time, 8-16
Shift Registers, 8-24, 11-24
in stages, 12-21
Shifting Accumulator Bits, 11-42
Sinking Circuits, 5-2
Sourcing Circuits, 5-2
Special Registers, 8-24
Special Relays, 8-24
using the pause relay, 13-13
Specifications
base power, 2-6
CPU, 3-3
discrete input modules, 6-4-6-15
discrete output modules, 7-6-7-20
environmental, 2-6
panel design, 2-4
power source, 2-6
Stages, 8-23
activating, 10-8
activating with power flow, 10-11
execution rules, 10-7
flowchart view, 10-26
instructions, 12-3
numbering, 10-6
parallel branching concepts, 10-19
resetting stages, 12-11
setting, 10-10
setting stages, 12-11
unusual operations, 10-24
using bits as contacts, 10-22, 12-7-12-10
using comparative boolean in, 10-18
using data instructions in, 10-17
using initial, 10-8
using outputs in, 10-12
using to latch outputs, 10-14
using with jump instructions, 10-9
Storing Programs, 3-4-3-8
Subtracting Numbers, 11-36
Switches, CPU
DL330/DL330P, 3-9
DL340, 3-10

DL340 communication, 3-12
System
component dimensions, 2-7
components, 1-4
enclosures, 2-7
environmental specifications, 2-6
operation, 8-2-8-37
panel design specifications, 2-4
power supply requirements, 2-6

## T

Terminology
discrete input, 6-2
discrete output, 7-2
Timers, 8-22, 9-8, 11-22
in stages, 10-15, 12-18
Troubleshooting
See also Indicators
I/O modules, 13-10
noise problems, 13-11
programs, 13-12
testing outputs, 13-8

UVPROM, 3-5
comparing to the CPU, 3-8
copying CPU program to, 3-7
erasing, 3-8
installing in CPU, 3-9, 3-10
loading program to CPU, 3-8

W
Weights, D-2
Wiring
base power, 2-11
bases, 4-7-4-9
I/O guidelines, 5-11
I/O modules, 2-12
run relay, 4-12
Word Memory, 8-19


[^0]:    *T/ C Setpoint Unit Only. Can be used as data registers if the Timer/Counter Setpoint Unit or Thumbwheel Interface Module is not used. R564 - R573 contain the preset value used with the Timer/ Counter Setpoint Unit. R674 - R677 contain the current values for these timers or counters.

[^1]:    * Control relays 340-373 can be made retentive by setting a CPU dipswitch. See Chapter 3 for details on setting CPU dipswitches.

[^2]:    * Control relays 340-373 can be made retentive by setting a CPU dipswitch. See Chapter 3 for details on setting CPU dipswitches.

