8

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

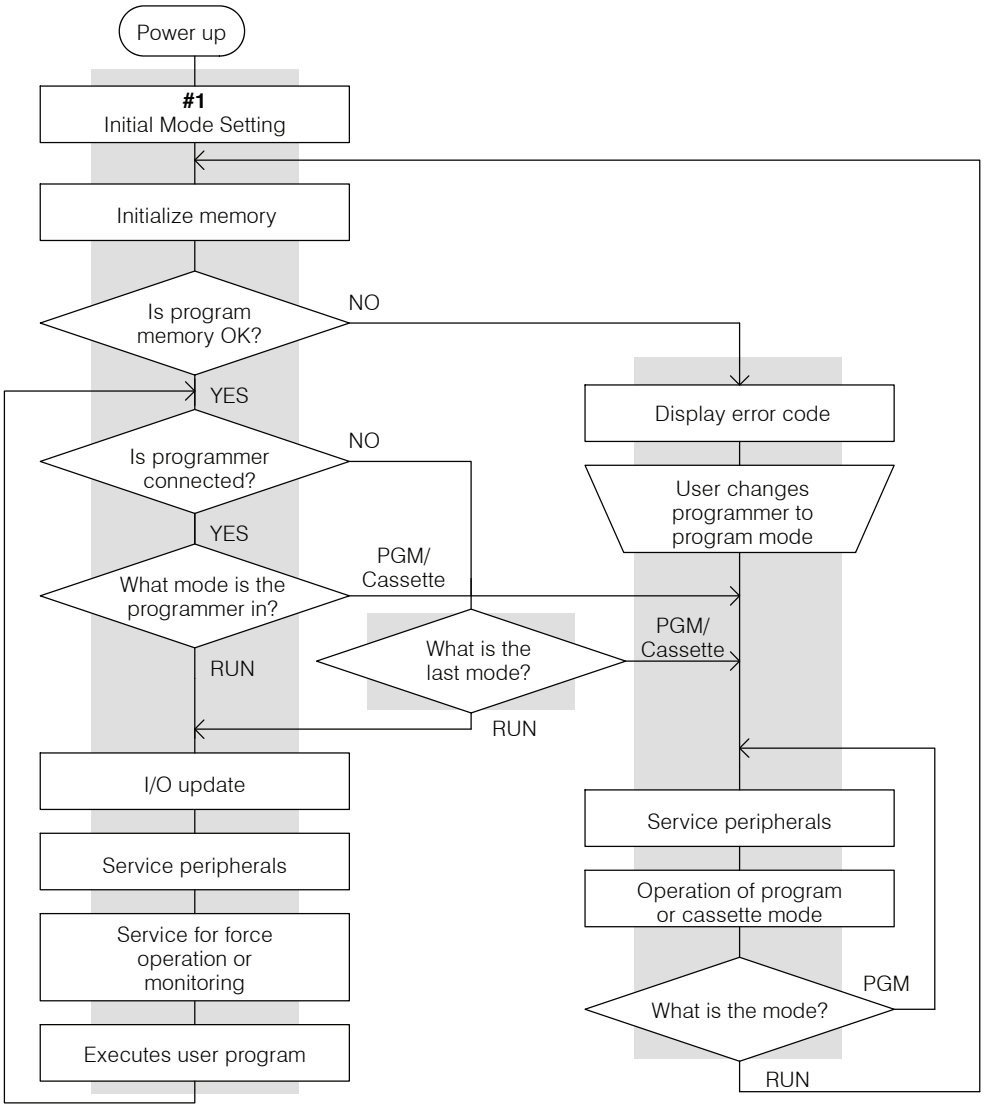
- **CPU Operating System** — the CPU is the heart of the system. It manages all aspects of system control. A quick overview of all the steps is provided in the next section.
- **CPU Operating Modes** — the CPU has different operating modes that allow different types of operations. There are two primary 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.

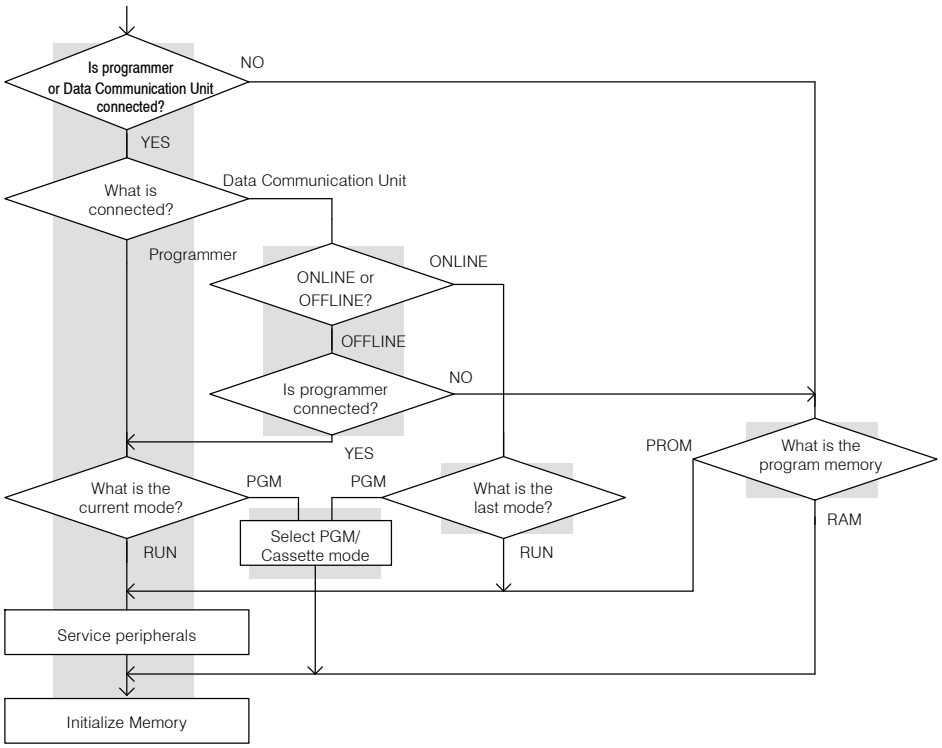
DL305 CPU Operational Flow Chart



Initial Mode Setting and Memory Initialization

Flow Chart for Initial Mode Setting (#1)

The previous flowchart contained a step called Initial Mode Setting. Once power has been connected to the system, the CPU executes the following procedure to 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.

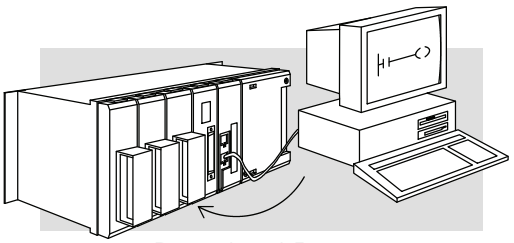
Item	Procedure
I/O Points	Input Points: activated, CPU will monitor status. Output Points: turned off
Control Relays	Non-Retentive: turned off Retentive: retains the condition prior to the system power interruption.
Shift Registers	Retains the condition prior to the system power interruption.
Timer /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



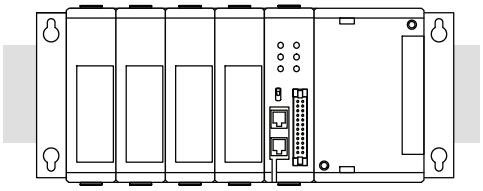
Download Program

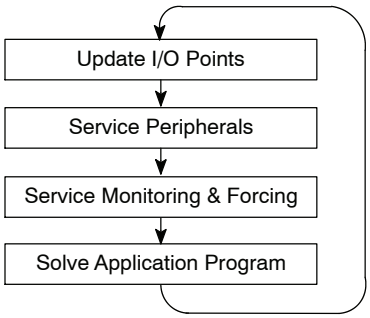
Run Mode Operation

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.



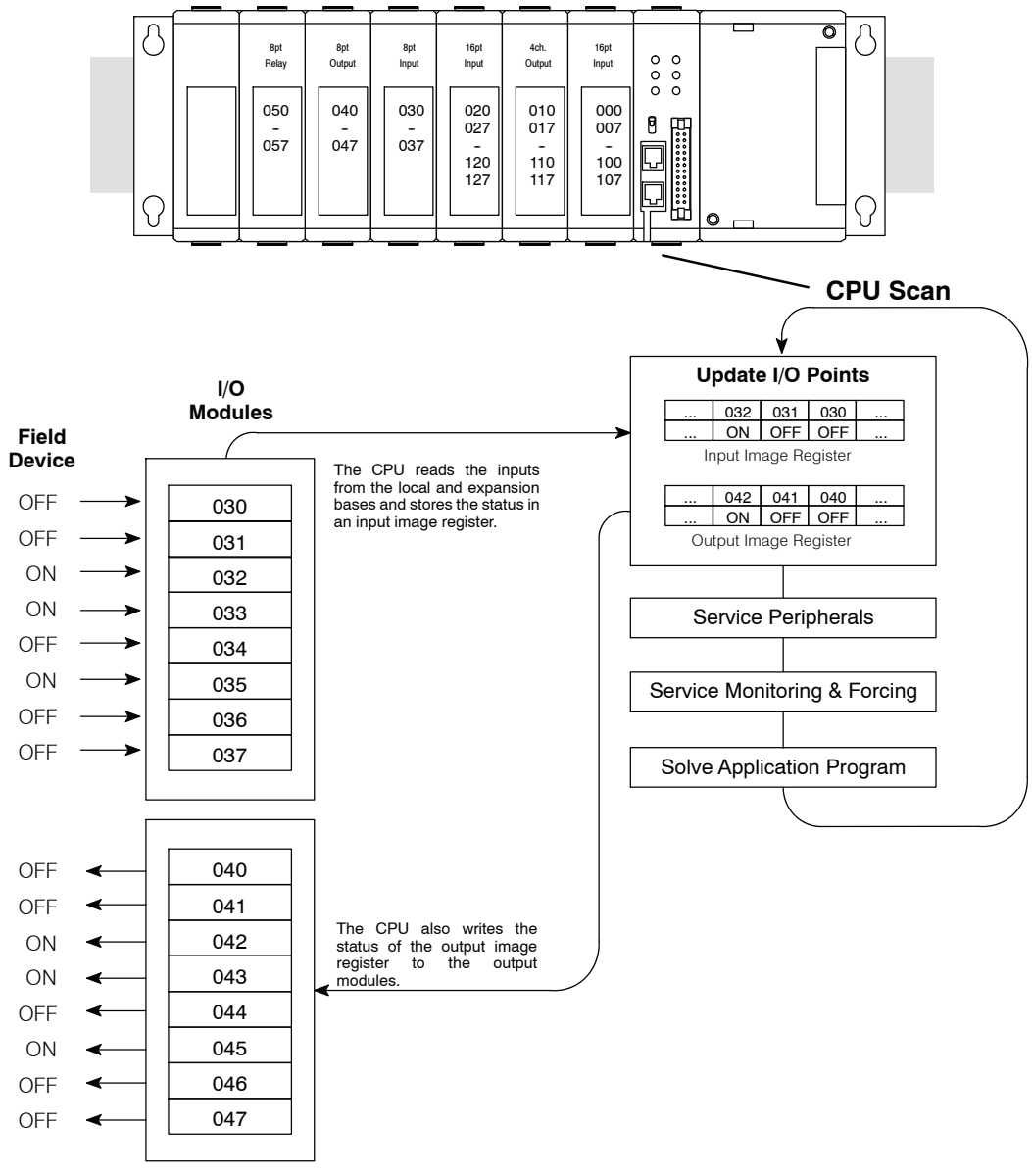


Update I/O Points

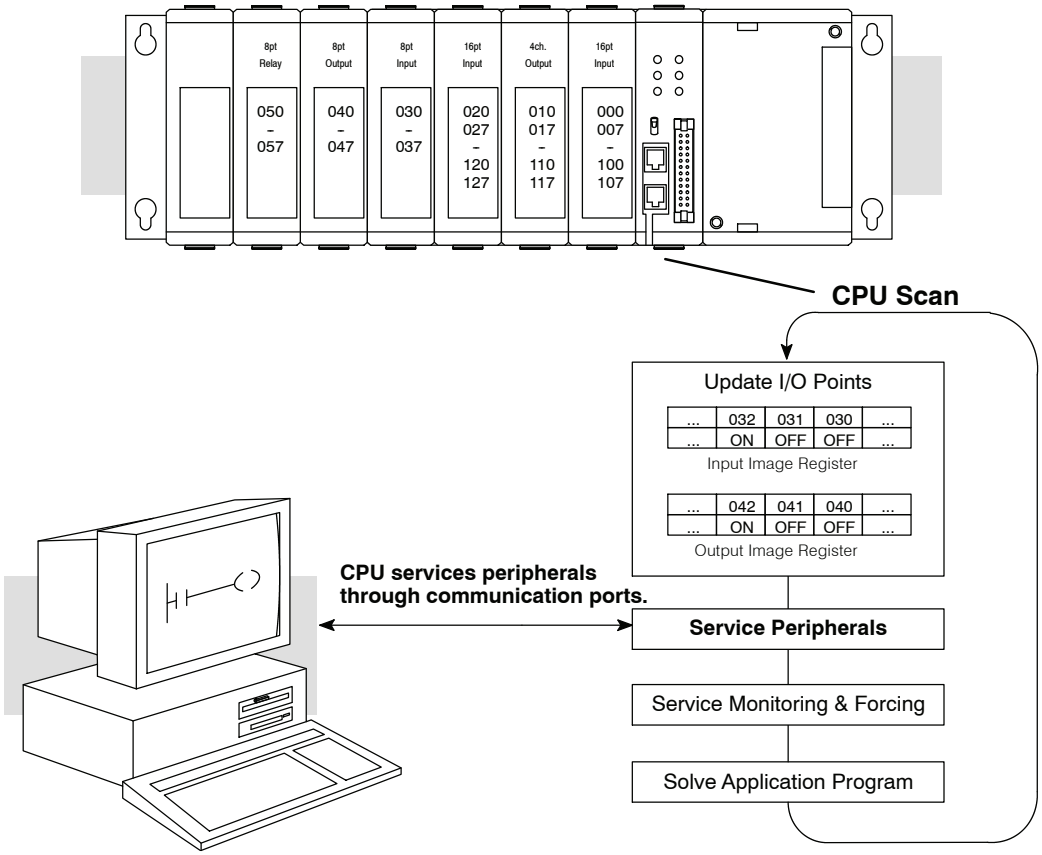
The CPU reads the status of all input modules present in the local CPU base and local expansion bases. The status of each input is stored in the input image register area. The input image register data is used by the CPU when it solves the application program.

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.



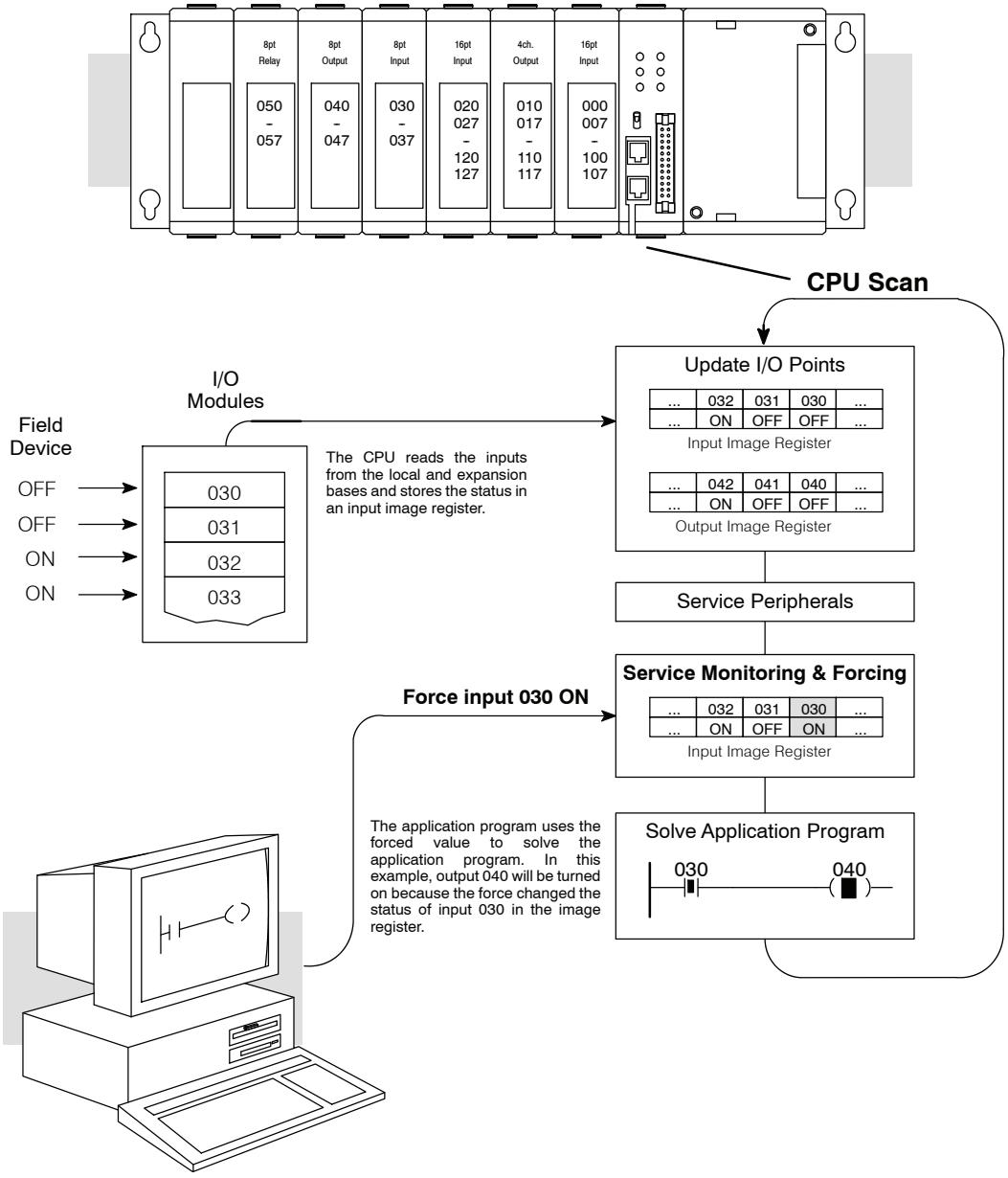
Service Peripherals After the CPU updates the I/O points, it reads any attached peripheral devices. This is primarily a communications service for any attached devices. For example, if you were using an operator interface to read or write data, the CPU would service these requests during this portion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



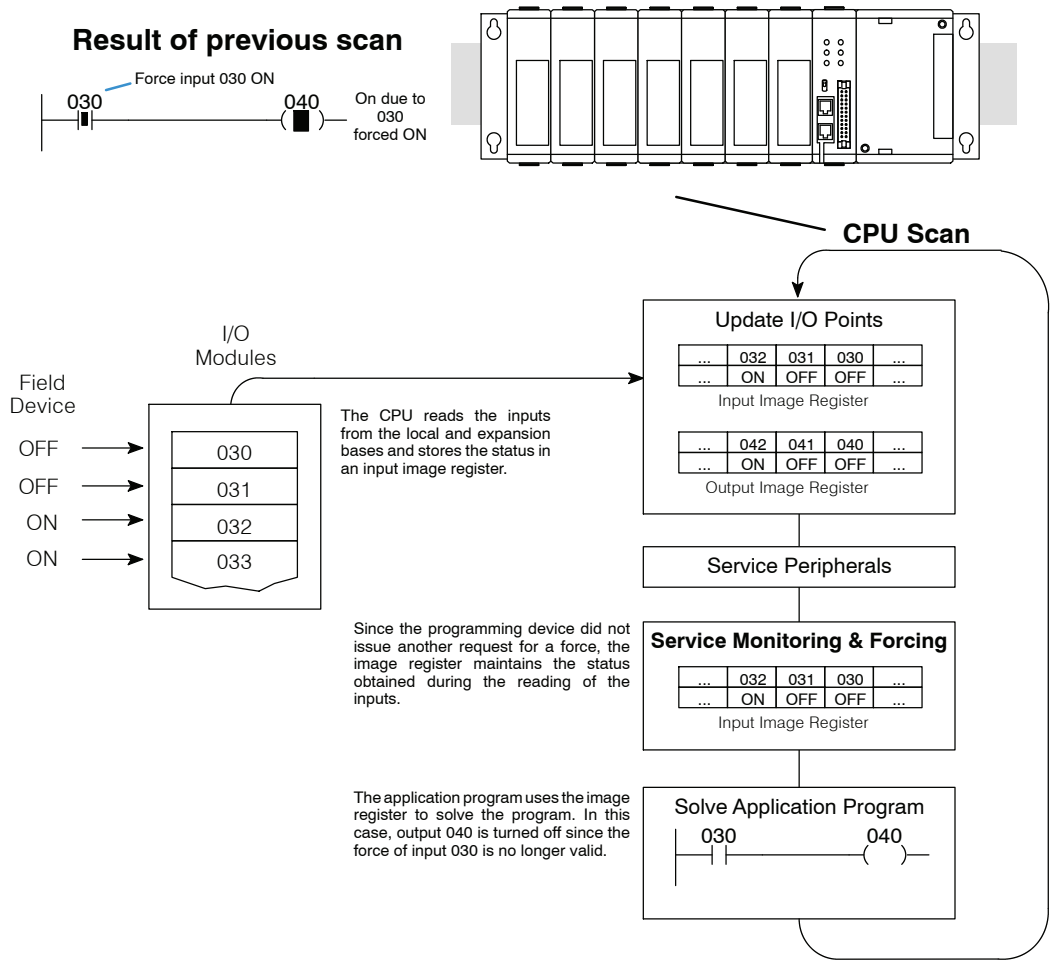
Service for Monitoring and Forcing Operations

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 Program

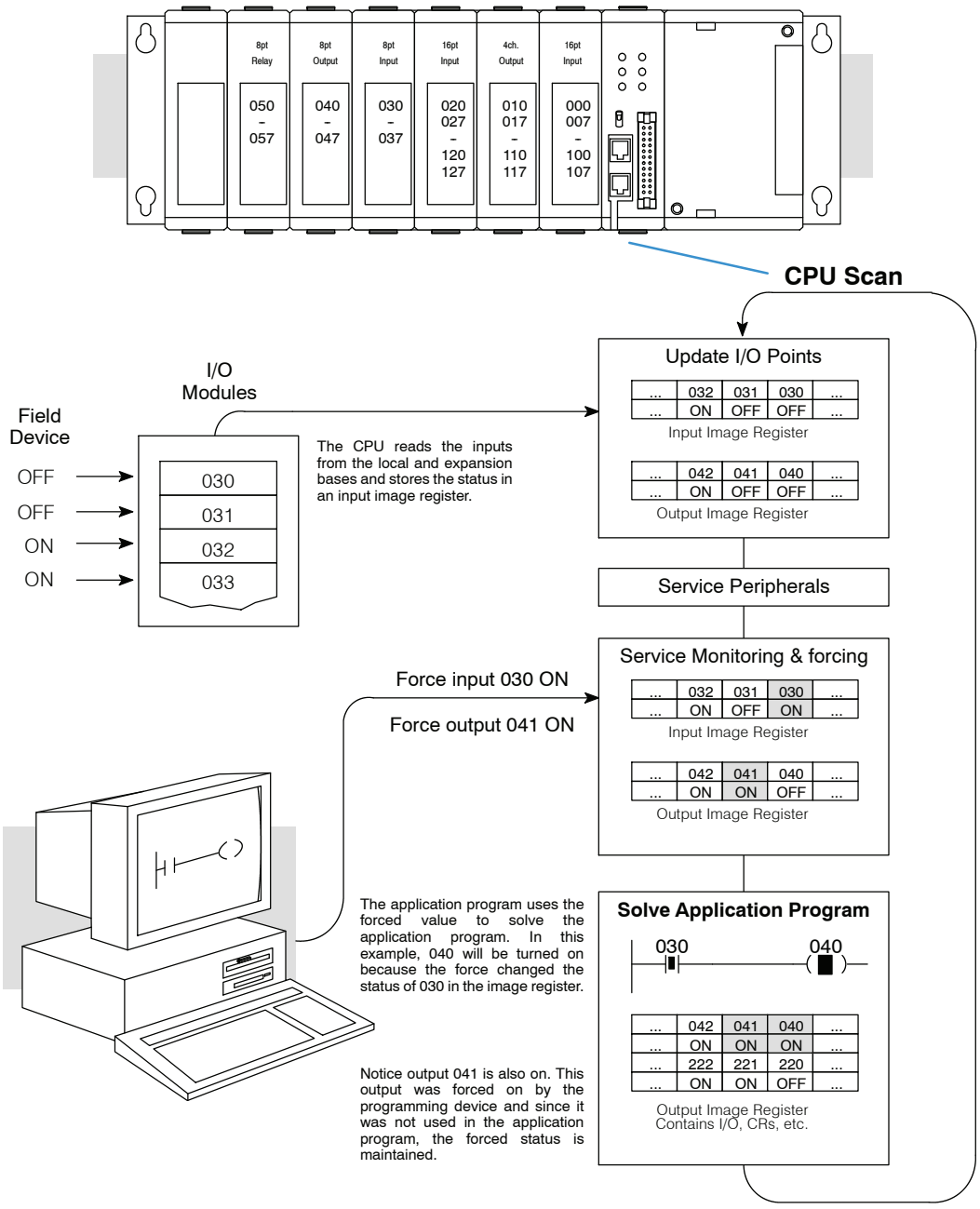
The CPU evaluates each instruction in the application program during this segment of the cycle. The instructions define the relationship between the input conditions and the desired output response.

The CPU uses the output image register area to store the status of the desired action for the outputs. The actual outputs are updated during the Update I/O Points segment of the cycle.

The internal control relays (C), stages (S), and data registers (R) are also updated in this segment.

You may recall the CPU may have obtained and stored forcing information when it serviced any peripheral devices. If any output points or register locations have been forced, the output image register also contains this information.

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.


8pt Relay
050 – 057
8pt Output
040 – 047
8pt Input
030 – 037
16pt Input
020 027 – 120 127
4ch. Output
010 017 – 110 117
16pt Input
000 007 – 100 107
CPU Scan
Update I/O Points
... 032 031 030 ...
... ON OFF OFF ...
Input Image Register
... 042 041 040 ...
... ON OFF OFF ...
Output Image Register
I/O Modules
Field Device
OFF
OFF
ON
ON
030
031
032
033
The CPU reads the inputs from the local and expansion bases and stores the status in an input image register.
Service Peripherals
Service Monitoring & forcing
Force input 030 ON
Force output 041 ON
... 032 031 030 ...
... ON OFF ON ...
Input Image Register
... 042 041 040 ...
... ON ON OFF ...
Output Image Register
The application program uses the forced value to solve the application program. In this example, 040 will be turned on because the force changed the status of 030 in the image register.
Solve Application Program
030
040
... 042 041 040 ...
... ON ON ON ...
... 222 221 220 ...
... ON ON OFF ...
Output Image Register
Contains I/O, CRs, etc.
Notice output 041 is also on. This output was forced on by the programming device and since it was not used in the application program, the forced status is maintained.

I/O Response Time

Is Timing Important for Your Application?

I/O response time is the amount of time required for the control system to sense a 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.

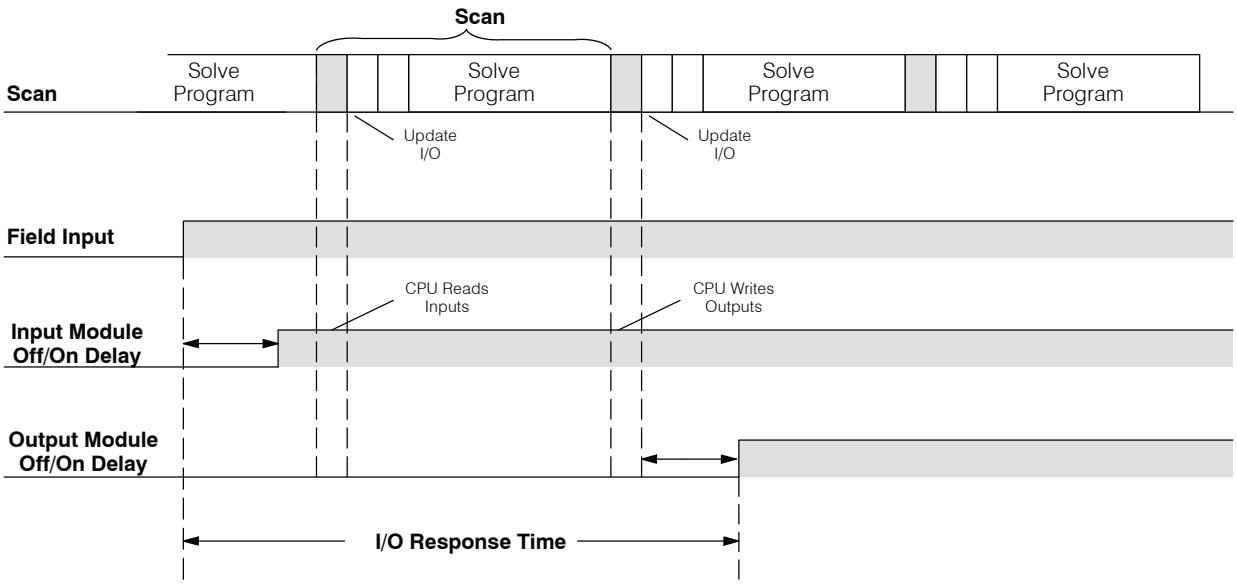
There are four things that can affect the I/O response time.

- 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

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.

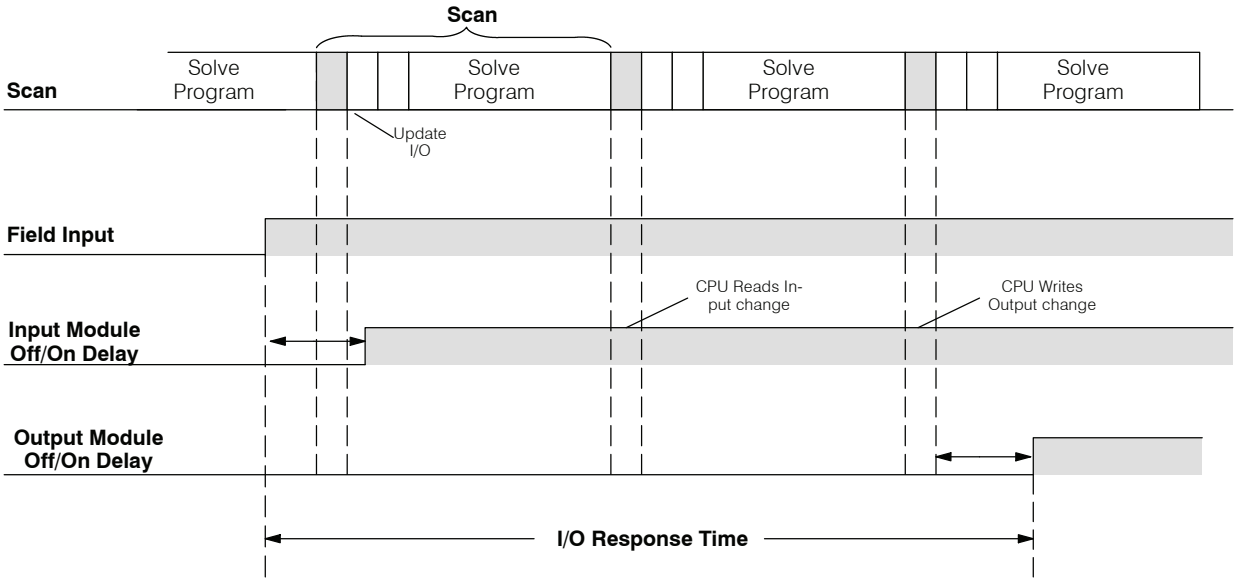


In this case, you can calculate the response time by simply adding the following items.

Input Delay + Scan Time + Output Delay = Response Time

Normal Maximum I/O Response

The I/O response time is longest when the module senses the input change just after 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 x Scan Time) + Output Delay = Response Time

Improving Response Time

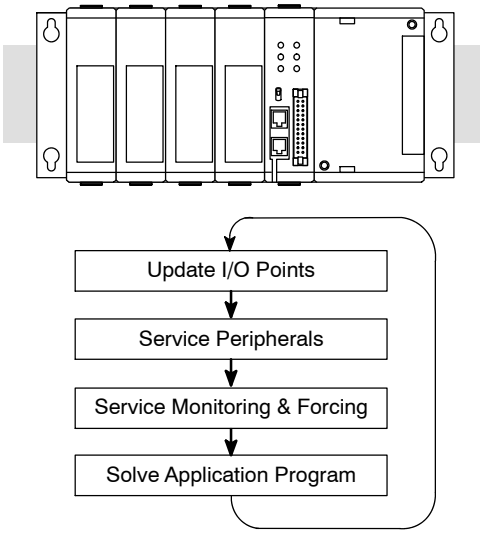
There are a few things you can do the help improve throughput.

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

CPU Scan Time Considerations

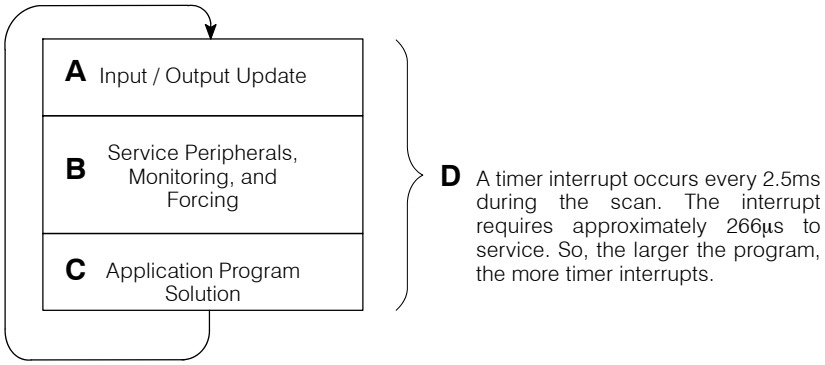
The scan time covers all the cyclical tasks that are performed by the operating system. This information can be very important when evaluating the performance of a system.

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.



DL330 / DL330P Scan Calculation

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



A = 3.3 ms typical I/O update

B = 0 - 5.2 ms maximum to service peripherals, monitoring, and forcing

C = Total of instruction execution time

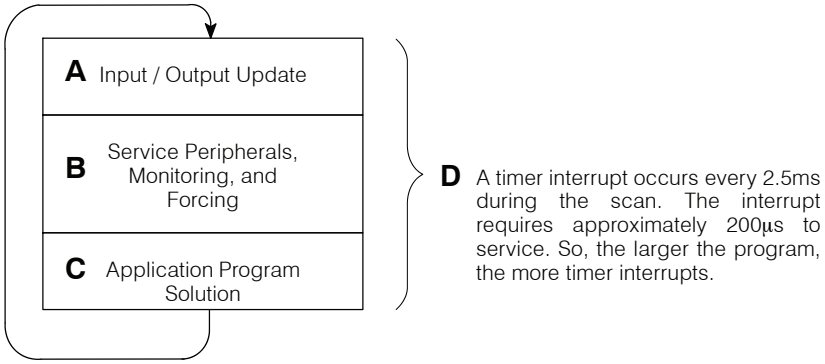
D = $266\mu s \times \frac{A + B + C + D}{2.5ms}$

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 – 3.5ms. However, the following table provides more precise execution timing guidelines for the execution cycle.



A = 2 ms typical I/O update

B = 0 – 1.2 ms maximum to service peripherals, monitoring, and forcing

C = Total of instruction execution time

D = $200\,\mu s \times \frac{A + B + C + D}{2.5ms}$

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.

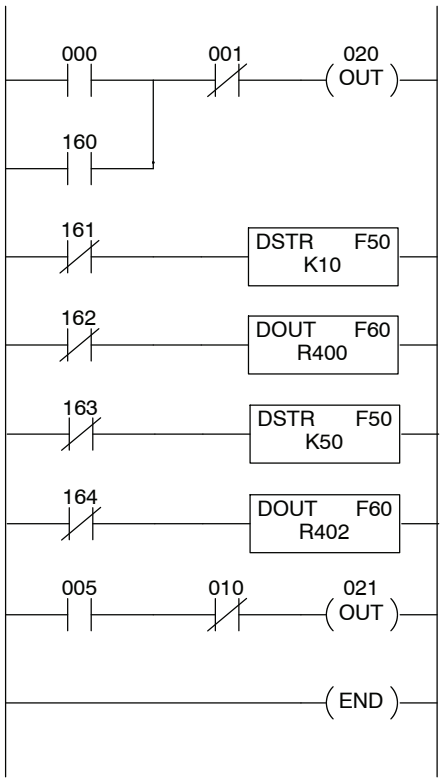
Application Program Execution

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	6.6µs
OR 160	6.6µs
ANDN 001	8.4µs
OUT 020	7.5µs
STRN 161	9.1µs
DSTR K10	14.3µs
STRN 162	9.1µs
DOUT R400	52.6µs
STRN 163	9.1µs
DSTR K50	14.3µs
STRN 164	9.1µs
DOUT R402	52.6µs
STR 005	6.6µs
ANDN 010	8.4µs
OUT 021	7.5µs
END	~0µs
TOTAL	221.8µs

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

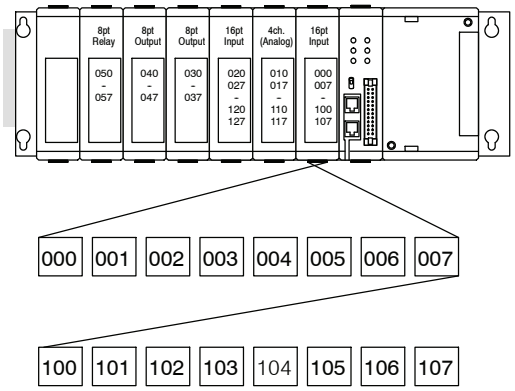
Memory Map

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.

Octal Numbering System

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.

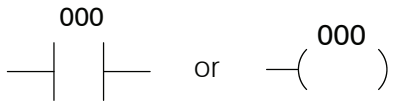


Two Basic Memory Types: Discrete and Word

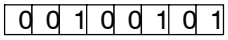
As you examine the different memory types, you'll notice two types of memory in the DL305, discrete and word memory. Discrete memory is one bit that can be either a 1 or a 0. Word memory is referred to as Data Register memory and is an 8-bit location normally used to manipulate data/numbers, store data/numbers, 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

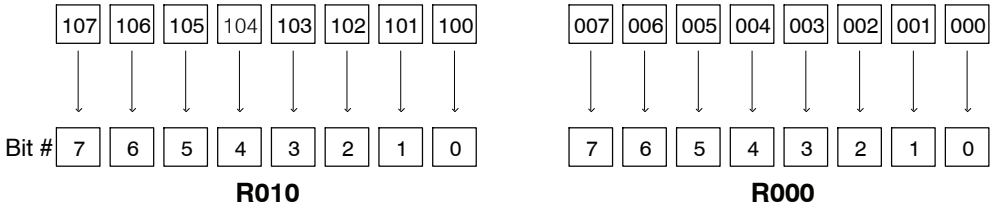


R Memory Locations for Discrete Memory Areas

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



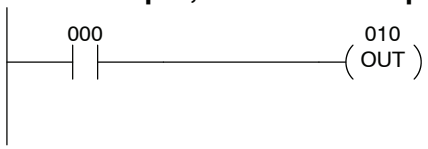
These discrete memory areas and their corresponding R-memory ranges are listed in the memory area tables at the end of this chapter.

I/O Points

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.

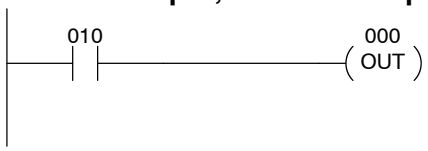
In the first example, output point 010 will be turned on when input 000 energizes. This assumes an 8-point input module is installed in the first slot and an 8-point output module is installed in the second slot.

1st Slot – Input, 2nd Slot – Output



1st Slot – Output, 2nd Slot – Input

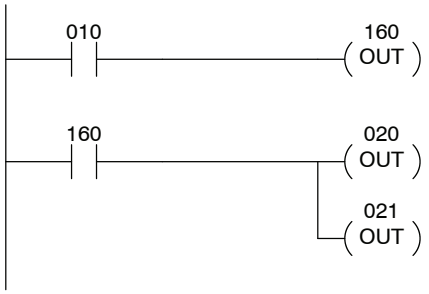
The second example shows how the numbers can represent a different type of point. For this example, the module positions were reversed.



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.

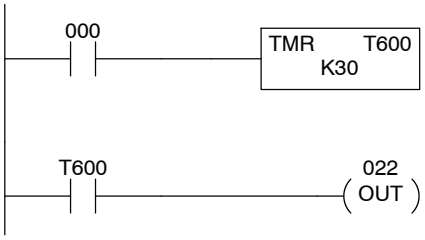
NOTE: Some of the references normally assigned as Control Relays can also be used to refer to a16-point I/O modules in some situations. Make sure you review the memory maps at the end of this chapter if you use 16-point modules.

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.

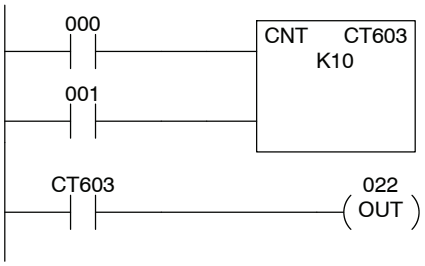


Counters and Counter Status Bits (CT Data type)

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 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.

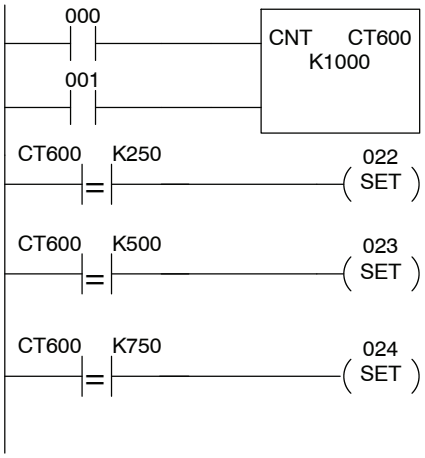
In the example shown, Each time contact 000 transitions from off to on, the counter increments by one. (If 001 comes on, the counter is reset to zero.) When the counter reaches the preset of 10 counts (K of 10) counter status contact CT603 turns on. When CT603 turns on, output 022 turns on.



Counter Current Values (R Data Type)

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.

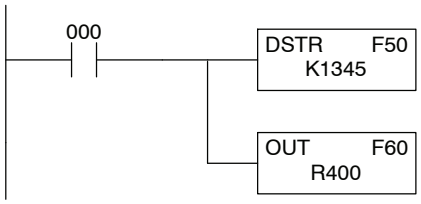


Data Registers (R Data Type)

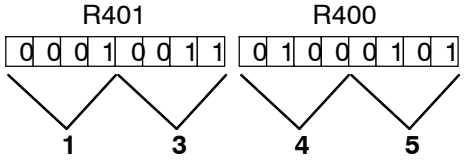
A word memory location is referred to as a Data Register, which is an 8-bit location normally used to manipulate data/numbers, store data/numbers, etc.

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.



Word Locations - 8 bits

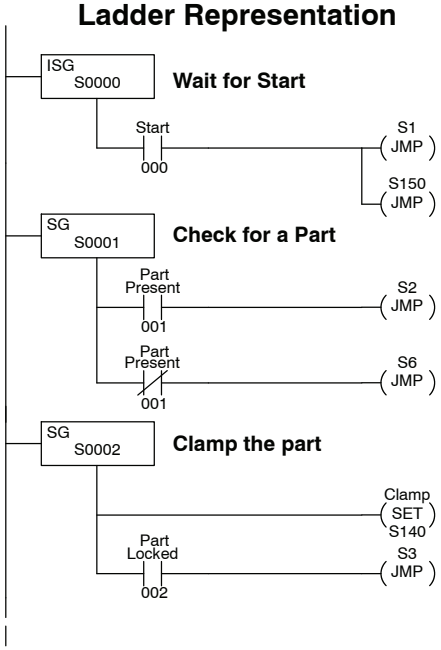


Stages (S Data type)

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 RLL*PLUS* 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

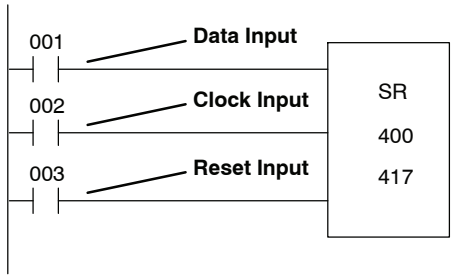


Shift Registers

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 completely separate areas and *are not* related.

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 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.



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 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)

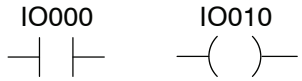
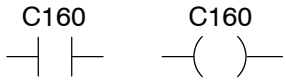
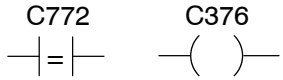
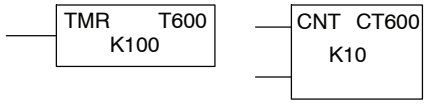
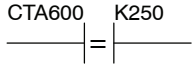
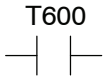
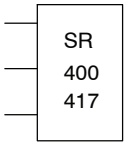
There are also a few special registers that store various types of system information. For example, the DL340 communication port parameters are set in special registers. The detailed memory maps at the end of this chapter show special register assignments for each CPU.

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.

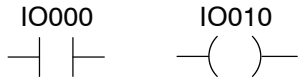
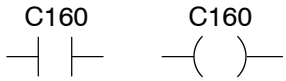
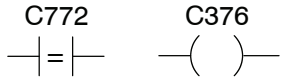
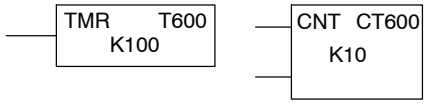
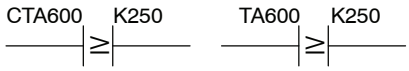
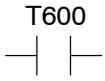
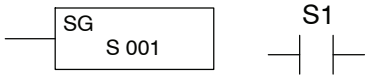
Address Start	Alias Start	Example
R0	RIO0	R0 is the 8-bit word memory reference for discrete bits IO0 through IO7, hence its alias is RIO0. R1 is the 8-bit word memory reference for discrete bits IO10 through IO17, hence its alias is RIO10.
R16	RC160	R16 is the 8-bit word memory reference for control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

DL330 Memory Map

Memory Type	Discrete Memory Reference (octal)	Register Memory Reference (octal)	Qty. Decimal	Symbol
Input / Output Points	000 – 157 700 – 767	R000 – R015 R070 – R076	168 Total	
Control Relays	160 – 373	R016 – R037	140	
Special Relays	374 – 377 770 – 777	R037 R077	12	
Timers / Counters	600 – 673 674 – 677*	None	64	
Timer / Counter Current Values	None	R600 – R673 R674 – R677*	64	 Contact valid for counters only.
Timer / Counter Status Bits	T600 – T673 T674 – T677*	None	64	
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 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. R564 – R573 contain the preset value used with the Timer / Counter Setpoint Unit. R674 – R677 contain the current values for these timers or counters.

DL330P Memory Map

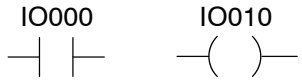
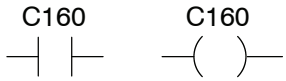
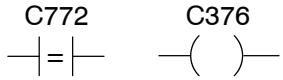
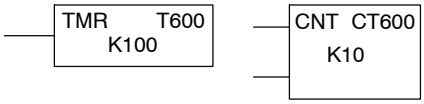
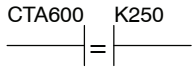
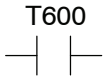
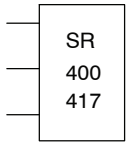
Memory Type	Discrete Memory Reference (octal)	Register Memory Reference (octal)	Qty. Decimal	Symbol
Input / Output Points	000 – 157 700 – 767	R000 – R015 R070 – R076	168 Total	
Control Relays	160 – 174 200 – 277	R016 – R017 R020 – R027	77	
Special Relays	175 – 177 770 – 777	R017 R077	11	
Timers / Counters	600 – 673 674 – 677*	None	64	
Timer / Counter Current Values	None	R600 – R673 R674 – R677*	64	
Timer / Counter Status Bits	T600 – T673 T674 – T677*	None	64	
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 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.

DL340 Memory Map

Memory Type	Discrete Memory Reference (octal)	Register Memory Reference (octal)	Qty. Decimal	Symbol
Input / Output Points	000 – 157 700 – 767	R000 – R015 R070 – R076	168 Total	
Control Relays	160 – 373 1000 – 1067	R016 – R037 R100 – R106	180	
Special Relays	374 – 377 770 – 777 1070 – 1077	R037 R077 R107	20	
Timers / Counters	600 – 673 674 – 677*	None	64	
Timer / Counter Current Values	None	R600 – R673 R674 – R677*	64	 Contact valid for counters only.
Timer / Counter Status Bits	T600 – T673 T674 – T677*	None	64	
Data Words	None	R400 – R563 R700 – R767	172	None specific, used with many instructions
Shift Registers	400 – 577	None	128	
Special Registers	None	R574 – R577 R770 – R777	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				LSB	Register Number
007	006	005	004	003	002	001	000	R0
017	016	015	014	013	012	011	010	R1
027	026	025	024	023	022	021	020	R2
037	036	035	034	033	032	031	030	R3
047	046	045	044	043	042	041	040	R4
057	056	055	054	053	052	051	050	R5
067	066	065	064	063	062	061	060	R6
077	076	075	074	073	072	071	070	R7
107	106	105	104	103	102	101	100	R10
117	116	115	114	113	112	111	110	R11
127	126	125	124	123	122	121	120	R12
137	136	135	134	133	132	131	130	R13
147	146	145	144	143	142	141	140	R14
157	156	155	154	153	152	151	150	R15
167	166	165	164	163	162	161	160	n/a
177	176	175	174	173	172	171	170	n/a
707	706	705	704	703	702	701	700	R70
717	716	715	714	713	712	711	710	R71
727	726	725	724	723	722	721	720	R72
737	736	735	734	733	732	731	730	R73
747	746	745	744	743	742	741	740	R74
757	756	755	754	753	752	751	750	R75
767	766	765	764	763	762	761	760	R76

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 ***Direct***SOFT.

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.

MSB			DL330 Control Relay References				LSB	Register Number
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
				373	372	371	370	R37

* Control relays 340 – 373 can be made retentive by setting a CPU dipswitch. See Chapter 3 for details on setting CPU dipswitches.

MSB			DL330P Control Relay References				LSB	Register Number
167	166	165	164	163	162	161	160	R16
			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

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

MSB			DL340 Control Relay References				LSB	Register Number
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
				373*	372	371	370	R37
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

* Control relays 340 – 373 can be made retentive by setting a CPU dipswitch. See Chapter 3 for details on setting CPU dipswitches.

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, ***Direct***SOFT 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.
DL330 DL340	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.
DL330 DL330P DL340	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. ***Direct***NET supports both ASCII and HEX modes and Modbus® 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*	676*	675*	674*	673	672	671	670

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

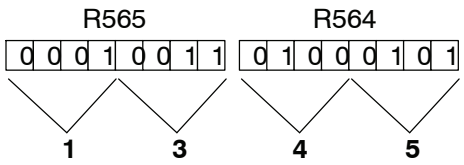
External Timer/Counter Setpoint Unit

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 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
				563	562	561	560

DL340 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
				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.

MSB			Stage References				LSB	Register Number
007	006	005	004	003	002	001	000	R100
017	016	015	014	013	012	011	010	R101
027	026	025	024	023	022	021	020	R102
037	036	035	034	033	032	031	030	R103
047	046	045	044	043	042	041	040	R104
057	056	055	054	053	052	051	050	R105
067	066	065	064	063	062	061	060	R106
077	076	075	074	073	072	071	070	R107
107	106	105	104	103	102	101	100	R110
117	116	115	114	113	112	111	110	R111
127	126	125	124	123	122	121	120	R112
137	136	135	134	133	132	131	130	R113
147	146	145	144	143	142	141	140	R114
157	156	155	154	153	152	151	150	R115
167	166	165	164	163	162	161	160	R116
177	176	175	174	173	172	171	170	R117

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						LSB	Register Number
407	406	405	404	403	402	401	400	R40
417	416	415	414	413	412	411	410	R41
427	426	425	424	423	422	421	420	R42
437	436	435	434	433	432	431	430	R43
447	446	445	444	443	442	441	440	R44
457	456	455	454	453	452	451	450	R45
467	466	465	464	463	462	461	460	R46
477	476	475	474	473	472	471	470	R47
507	506	505	504	503	502	501	500	R50
517	516	515	514	513	512	511	510	R51
527	526	525	524	523	522	521	520	R52
537	536	535	534	533	532	531	530	R53
547	546	545	544	543	542	541	540	R54
557	556	555	554	553	552	551	550	R55
567	566	565	564	563	562	561	560	R56
577	576	575	574	573	572	571	570	R57

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

Special Registers

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

CPUs	Special Register	Description of Contents
DL330 DL330P DL340	R574 – 575	Contains the error code used with the FAULT instruction.
	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.