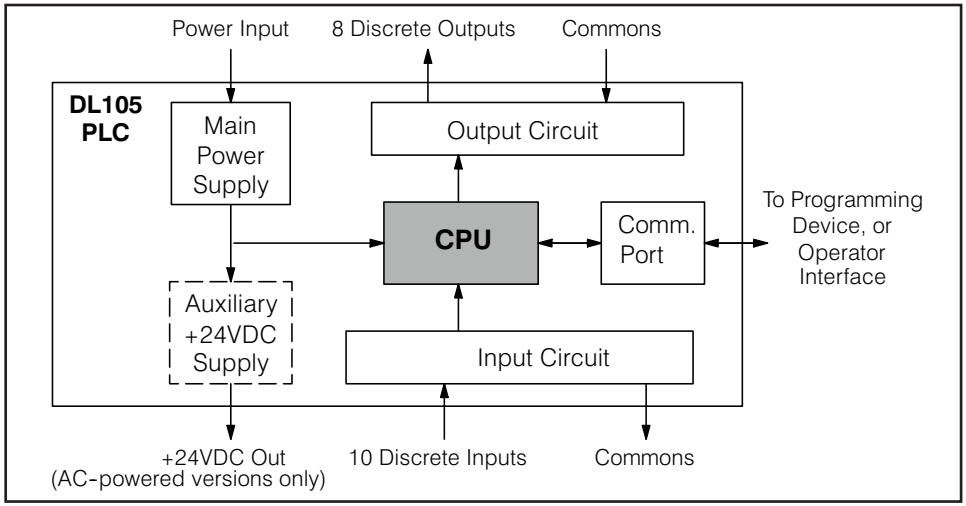
CPU Specifications and Operation

In This Chapter...

Introduction

The Central Processing Unit (CPU) is the heart of the Micro PLC. Almost all PLC operations are controlled by the CPU, so it is important that it is set up correctly. This chapter provides the information needed to understand:

- Steps required to set up the CPU
- Operation of ladder program, organization of Variable Memory



NOTE: *The High-Speed I/O function (HSIO) consists of dedicated but configurable hardware in the DL105. It is not considered part of the CPU, because it does not execute the ladder program. For more on HSIO operation, see Chapter 3.*

DL105 CPU Features

The DL105 Micro PLC which has 2.4K words of memory comprised of 2.0K of ladder memory and 384 words of V-memory (data registers). Program storage is in the FLASH memory which is a part of the CPU board in the PLC. In addition, there is RAM with the CPU which will store system parameters, V-memory, and other data which is not in the application program. The RAM is backed up by a "super-capacitor", storing the data for several days in the event of a power outage. The capacitor automatically charges during powered operation of the PLC.

The DL105 supports fixed I/O which includes ten discrete input points and eight output points. No provision for expansion beyond these eighteen I/O points are available in the F1-130 model PLCs.

Over 90 different instructions are available for program development as well as extensive internal diagnostics that can be monitored from the application program or from an operator interface. Chapter 5 provides a detailed description of the instructions.

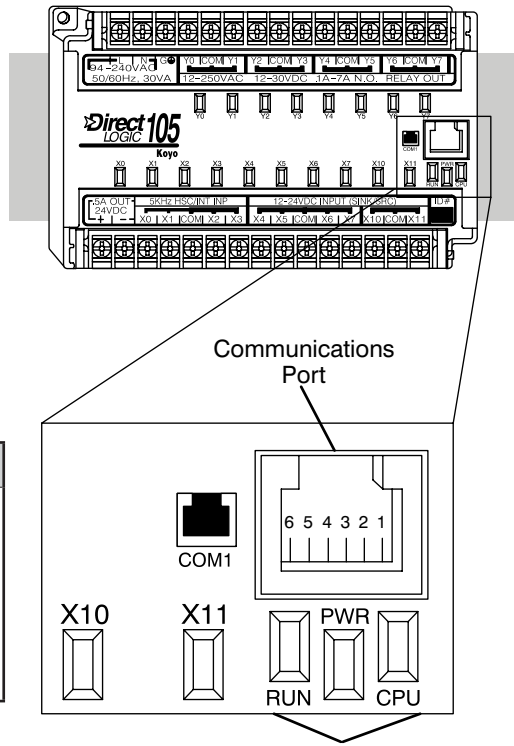
The DL105 provides one built-in RS-232C communication port, so you can easily connect a handheld programmer or a personal computer without needing any additional hardware.

CPU Specifications

Feature	DL105
Total Program memory (words)	2.4K
Ladder memory (words)	2048
Total V- memory (words) (See Appendix E)	384
User V- memory (words)	256
Non-volatile V- Memory (words)	128
Contact execution (boolean)	3.3 µs
Typical scan (boolean)	4 to 6 ms
RLL Ladder style Programming	Yes
RLL and RLL^{PLUS} Programming	Yes
Run Time Edits	Yes
Variable / fixed scan	Variable
Supports Override	No
Handheld programmer	Yes
*Direct*SOFT programming for Windows™	Yes
Built-in communication ports (RS-232C)	Yes
EEPROM or FLASH	Standard on CPU
Local Discrete I/O points available	18
Local Analog input / output channels maximum	None
High-Speed I/O (quad, pulse out, interrupt, pulse catch, etc.)	Yes
I/O Point Density	10 inputs, 8 outputs
Number of instructions available (see Chapter 5 for details)	91
Control relays	256
Special relays (system defined)	112
Stages in RLL^{PLUS}	256
Timers	64
Counters	64
Immediate I/O	Yes
Interrupt input (external / timed)	Yes
Subroutines	No
For/Next Loops	No
Math	Integer
Drum Sequencer Instruction	Yes
Time of Day Clock/Calendar	No
Internal diagnostics	Yes
Password security	Yes
System error log	No
User error log	No
Battery backup	No (uses super-cap.)

CPU Hardware Setup

CPU Status Indicators		
RUN	ON	CPU is in RUN mode
	OFF	CPU is in Program mode
	FLASHING	CPU is in Upgrade mode
CPU	ON	CPU internal diagnostics has detected an error.
	OFF	CPU is OK.
PWR	ON	CPU power good
	OFF	CPU power failure

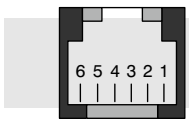


Communication Port	
Com1	Connects to HPP, *Direct*SOFT, operator interfaces, etc. 6-pin, RS-232C 9600 Baud Odd parity Station address fixed (1) 8 data bits 1 start, 1 stop bit Asynchronous, Half-duplex, DTE K sequence protocol

Communication Port Pinout Diagrams

Cables are available that allow you to quickly and easily connect a Handheld Programmer or a personal computer to the DL105 PLCs. However, if you need to build your own cables, use the pinout diagrams shown. The DL105 PLCs require an RJ-12 phone plug to fit the built-in jacks.

The Micro PLC has one built-in RS-232C communication port. The port is generally used for programming either with the Handheld Programmer or *Direct*SOFT, and has a fixed station address of 1. The baud rate is fixed at 9600 baud. This port supports the K- sequence protocol, which is a proprietary protocol.



Phone Jack Connector

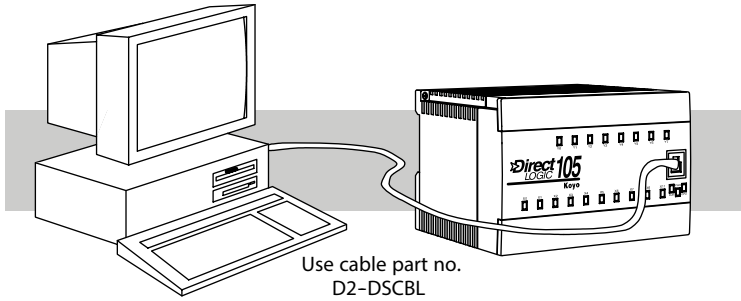
Port Pinouts

Pin	Signal Definition
1	0 V
2	5 V
3	RS232C Data in
4	RS232C Data out
5	5 V

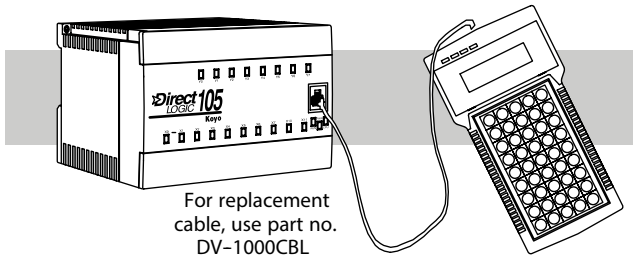
NOTE: *The 5V pins are rated at 200mA maximum, primarily for use with some operator interface units.*

Connecting the Programming Devices

If you're using a Personal Computer with the ***Direct***SOFT programming package, you can connect the computer to the DL105's programming port. For an engineering office environment (typical during program development), this is the preferred method of programming.



The Handheld programmer is connected to the CPU with a handheld programmer cable. This device is ideal for maintaining existing installations or making small program changes. The handheld programmer is shipped with a cable, which is approximately 6.5 feet (200 cm) long.



CPU Setup Information

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, but it also includes some things that you may never have to use. Here's a brief list of the items that are discussed.

- Using Auxiliary Functions
- Selecting and Changing the CPU Modes
- Clearing the program (and other memory areas)
- How to initialize system memory
- Setting retentive memory ranges

The following paragraphs provide the setup information necessary to get the CPU ready for programming. They include setup instructions for either type of programming device you are using. The D2- HPP Handheld Programmer Manual provides the Handheld keystrokes required to perform all of these operations. The ***Direct***SOFT Manual provides a description of the menus and keystrokes required to perform the setup procedures via ***Direct***SOFT.

CPU Modes

There are two possible operating modes available with DL105 Micro PLCs.

- RUN — executes program and updates I/O points.
- PROGRAM — allows program changes. The CPU halts execution of the ladder program and all output points are turned off.

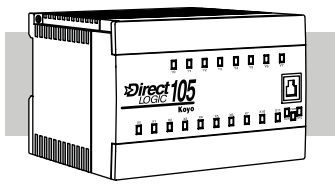
Mode of Operation at Power-up

The DL105 operates as follows when the power is connected.

1. The DL105 CPU will normally power-up in the mode that it was in just prior to the power interruption. For example, if the CPU was in Program Mode when the power was disconnected, the CPU will power-up in Program Mode (see warning note below).
2. You can configure the DL105 to always power-up in the Run Mode. You can set bit 13 in V7633 (nonvolatile memory) to enable this feature; we'll show you how to set the bit later in this chapter.

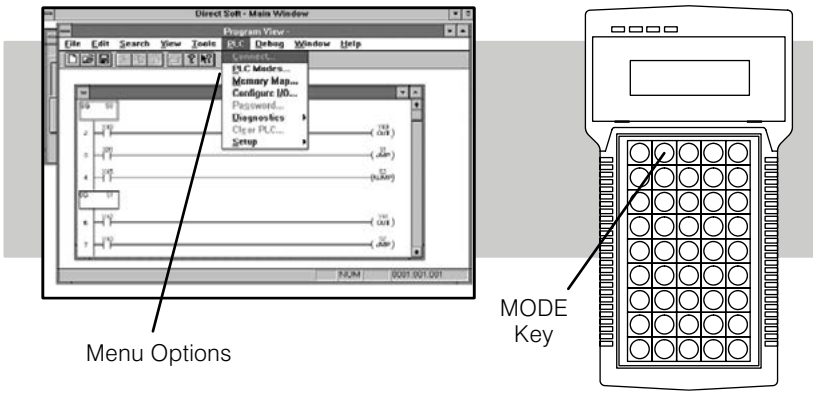
WARNING: If bit 13 in memory location V7633 is not set, once the super capacitor has discharged the system memory may not retain the previous mode of operation. When this occurs, the PLC can power-up in either Run or Program Mode. There is no way to determine which mode will be entered. Failure to adhere to this warning greatly increases the risk of unexpected equipment startup.

Changing Modes in the DL105 PLC



The DL105 Micro PLC does not have an external switch to switch CPU operating modes. You have to use a programming device, such as the handheld programmer or ***Direct***SOFT, to change the operating mode.

You can use either ***Direct***SOFT or the Handheld Programmer to change the CPU mode of operation. With ***Direct***SOFT you use a menu option in the PLC menu. With the Handheld Programmer, you use the MODE key.

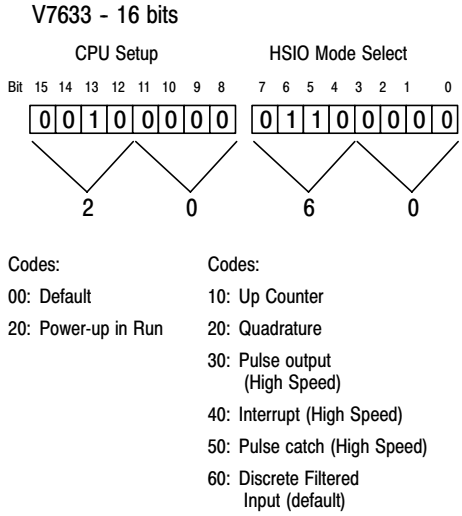


Setting Bits in V7633

You can use the Handheld Programmer or ***Direct***SOFT to set the proper bits in V7633.

Since you cannot access the bits individually, you have to enter a constant that will result in the appropriate bit being set. The first two digits of the constant are used to select the CPU options. The second two digits are used with the High-Speed I/O function to select various options. If you're using High Speed I/O functions, make sure you also enter the appropriate code for the feature selected If you want the HSIO inputs and outputs to default to regular I/O point operation, just enter 60 as the last two digits of the code. This configures all I/O points to operate only as standard discrete I/O.

The diagram shows how the upper and lower bytes of V7633 are used. For example, if you entered 2060 into V7633, the powerup-in-run option is selected, and the discrete filtered inputs are selected.



The following keystrokes show how you can enter the codes into V7633 with the D2-HPP Handheld Programmer.

Select V7633 for Monitoring

CLR CLR SHFT

V AND | H 7 | G 6 | D 3 | D 3 | STAT

```
V 7634  V7633
  0001    0000
```

Enter the Code

SHFT | K JMP | C 2 | A 0 | G 6 | A 0 | ENT

```
 V 7634  V7633
K2060
```

Since the changes take affect immediately, you may receive an error message. For example, if you select Power-up in Run Mode and the CPU does not yet contain a program, an error will occur.

Auxiliary Functions

Many CPU setup tasks involve the use of Auxiliary (AUX) Functions. The AUX Functions perform many different operations, ranging from clearing ladder memory, displaying the scan time, copying programs to EEPROM in the handheld programmer, etc. They are divided into categories that affect different system parameters. Appendix A provides a description of the AUX functions.

You can access the AUX Functions from ***Direct***SOFT or from the D2-HPP Handheld Programmer. The manuals for those products provide step-by-step procedures for accessing the AUX Functions. Some of these AUX Functions are designed specifically for the Handheld Programmer setup, so they will not be needed (or available) with the ***Direct***SOFT package. The following table shows a list of the Auxiliary functions for the Handheld Programmer.

Clearing an Existing Program

Before you enter a new program, be sure to always clear ladder memory. You can use AUX Function 24 to clear the complete program.

You can also use other AUX functions to clear other memory areas.

- AUX 23 — Clear Ladder Range
- AUX 24 — Clear all Ladders
- AUX 31 — Clear V-Memory

The DL105 Micro PLC maintain system parameters in a memory area often referred to as the "scratchpad". In some cases, you may make changes to the system setup that will be stored in system memory. For example, if you specify a range of Control Relays (CRs) as retentive, these changes are stored in system memory.

AUX 54 resets the system memory to the default values.

Initializing System Memory

WARNING: You may never have to use this feature unless you want to clear any setup information that is stored in system memory. Usually, you'll only need to initialize the system memory if you are changing programs and the old program required a special system setup. You can usually load in new programs without ever initializing system memory. Remember, this AUX function will reset all system memory. If you have set special parameters such as retentive ranges, etc. they will be erased when AUX 54 is used. Make sure you that you have considered all ramifications of this operation before you select it.

Setting Retentive Memory Ranges

The DL105 PLCs provide certain ranges of retentive memory by default. The default ranges are suitable for many applications, but you can change them if your application requires additional retentive ranges or no retentive ranges at all. (see Appendix E) The default settings are:

You can use AUX 57 (see Appendix A) to set the retentive ranges. You can also use ***Direct***SOFT menus to select the retentive ranges.

WARNING: The DL105 PLCs do not have battery back-up. The super capacitor will retain the values in the event of a power loss, but only for a short period of time, depending on conditions. If the retentive ranges are important for your application, make sure you program critical parameters into EEPROM locations.

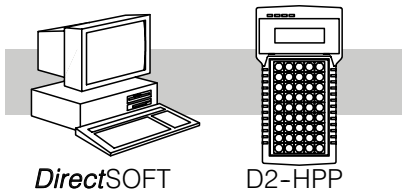
Using a Password

The DL105 PLCs allow you to use a password to help minimize the risk of unauthorized program and/or data changes. Once you enter a password you can "lock" the PLC against access. Once the CPU is locked you must enter the password before you can use a programming device to change any system parameters.

You can select an 8-digit numeric password. The Micro PLCs are shipped from the factory with a password of 00000000. All zeros removes the password protection. If a password has been entered into the CPU you cannot just enter all zeros to remove it. Once you enter the correct password, you can change the password to all zeros to remove the password protection.

WARNING: Make sure you remember your password. If you forget your password you will not be able to access the CPU. The Micro PLC must be returned to the factory to have the password removed.

You can use the D2-HPP Handheld Programmer or ***Direct***SOFT to enter a passward. The following diagram shows how you can enter a password with the Handheld Programmer.



Select AUX 81

PASSWORD
00000000

Enter the new 8-digit password

PASSWORD
XXXXXXXX

Press CLR to clear the display

There are three ways to lock the CPU once the password has been entered.

1. If the CPU power is disconnected, the CPU will be automatically locked against access.
2. If you enter the password with ***Direct***SOFT, the CPU will be automatically locked against access when you exit ***Direct***SOFT.
3. Use AUX 83 to lock the CPU.

When you use ***Direct***SOFT, you will be prompted for a password if the CPU has been locked. If you use the Handheld Programmer, you have to use AUX 82 to unlock the CPU. Once you enter AUX 82, you will be prompted to enter the password.

CPU Operation

Achieving the proper control for your equipment or process requires a good understanding of how DL105 CPUs control all aspects of system operation. There are four main areas to understand before you create your application program:

- CPU Operating System — the CPU manages all aspects of system control. A quick overview of all the steps is provided in the next section.
- CPU Operating Modes — The two primary modes of operation are Program Mode and Run Mode.
- CPU Timing — The two important areas we discuss are the I/O response time and the CPU scan time.
- CPU Memory Map — DL105 CPUs offer a wide variety of resources, such as timers, counters, inputs, etc. The memory map section shows the organization and availability of these data types.

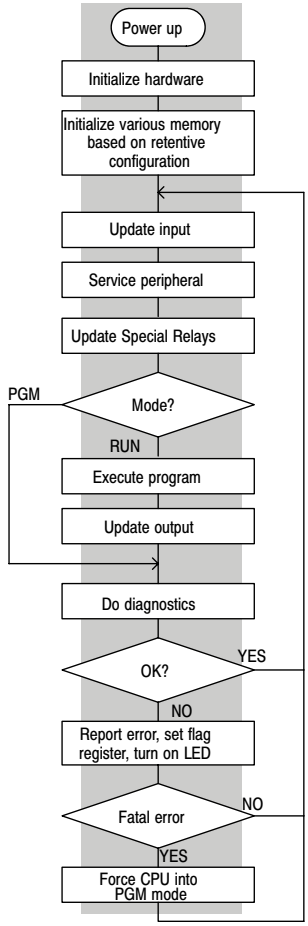
CPU Operating System

At powerup, the CPU initializes the internal electronic hardware. Memory initialization starts with examining the retentive memory settings. In general, the contents of retentive memory is preserved, and non-retentive memory is initialized to zero (unless otherwise specified).

After the one-time powerup tasks, the CPU begins the cyclical scan activity. The flowchart to the right shows how the tasks differ, based on the CPU mode and the existence of any errors. The "scan time" is defined as the average time around the task loop. Note that the CPU is always reading the inputs, even during program mode. This allows programming tools to monitor input status at any time.

The outputs are only updated in Run mode. In program mode, they are in the off state.

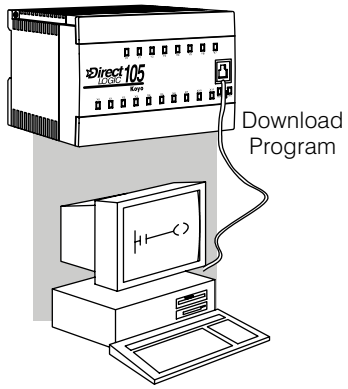
Error detection has two levels. Non-fatal errors are reported, but the CPU remains in its current mode. If a fatal error occurs, the CPU is forced into program mode and the outputs go off.



Program Mode

In Program Mode, the CPU does not execute the application program or update the output points. The primary use for Program Mode is to enter or change an application program. You also use program mode to set up the CPU parameters, such as HSIO features, retentive memory areas, etc.

You can use a programming device, such as ***Direct***SOFT or the D2- HPP Handheld Programmer to place the CPU in Program Mode.



Run Mode

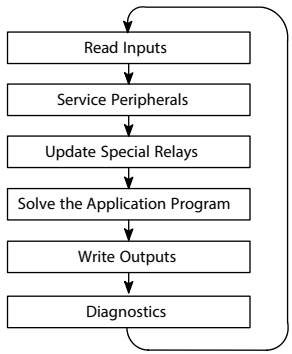
In Run Mode, the CPU executes the application program and updates the I/O system. You can perform many operations during Run Mode. Some of these include:

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

Run Mode operation can be divided into several key areas. For the vast majority of applications, some of these execution segments are more important than others. For example, you need to understand how the CPU updates the I/O points, handles forcing operations, and solves the application program. The remaining segments are not that important for most applications.

You can use ***Direct***SOFT or the D2- HPP Handheld Programmer to place the CPU in Run Mode.

Normal Run mode scan



You can also edit the program during Run Mode. The Run Mode Edits are not "bumpless" to the outputs. Instead, the CPU maintains the outputs in their last state while it accepts the new program information. If an error is found in the new program, then the CPU will turn all the outputs off and enter the Program Mode. This feature is discussed in more detail in Chapter 8.

WARNING: Only authorized personnel fully familiar with all aspects of the application should make changes to the program. Changes during Run Mode become effective immediately. Make sure you thoroughly consider the impact of any changes to minimize the risk of personal injury or damage to equipment.

Read Inputs

The CPU reads the status of all inputs, then stores it in the image register. Input image register locations are designated with an X followed by a memory location. Image register data is used by the CPU when it solves the application program.

Of course, an input may change after the CPU has just read the inputs. Generally, the CPU scan time is measured in milliseconds. If you have an application that cannot wait until the next I/O update, you can use Immediate Instructions. These do not use the status of the input image register to solve the application program. The Immediate instructions immediately read the input status directly from the I/O modules. However, this lengthens the program scan since the CPU has to read the I/O point status again. A complete list of the Immediate instructions is included in Chapter 5.

Service Peripherals and Force I/O

After the CPU reads the inputs from the input modules, it reads any attached peripheral devices. This is primarily a communications service for any attached devices. For example, it would read a programming device to see if any input, output, or other memory type status needs to be modified.

Forced I/O — temporarily changes the status of a discrete bit. 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 next scan. This is primarily useful during testing situations when you just need to force a bit on to trigger another event.

Forced Inputs — The CPU reads the status of X inputs during the Read Inputs portion of the scan. When the CPU services the programming device, it logs any request to force an X input on. If the input is used in the application program, the ladder X contact is considered closed (on). Since an X input is a real-world input point, the CPU will change the status when it reads the inputs on the next scan.

Forced Outputs — Outputs which are not used in the program can be forced on and off for troubleshooting and maintenance purposes. You can temporarily allow the forcing of any output by inserting an END coil instruction at the beginning of the ladder program. Then you can use ***Direct***SOFT or a HPP to force outputs on and off.

The DL105 PLCs only retain the forced value for one scan. There is an exception to this rule. For example, if the point address is greater than X11 or Y7 or it is not used in the ladder program, then the point will maintain the forced status.

WARNING: Only authorized personnel fully familiar with all aspects of the application should make changes to the program. Make sure you thoroughly consider the impact of any changes to minimize the risk of personal injury or damage to equipment.

Update Special Relays and Special Registers

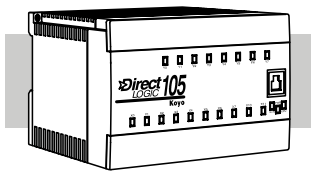
There are certain V-memory locations that contain Special Relays and other dedicated register information. This portion of the execution cycle makes sure these locations get updated on every scan. Also, there are several different Special Relays, such as diagnostic relays, etc., that are also updated during this segment.

Solve Application Program

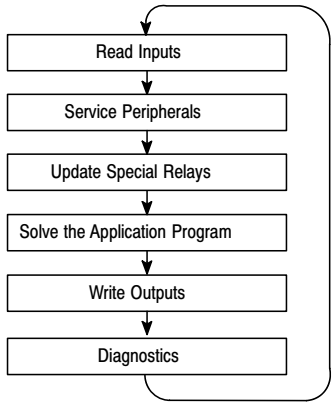
The CPU evaluates each instruction in the application program during this segment of the scan cycle. The instructions define the relationship between the input conditions and the desired output response. The CPU uses the output image register area to store the status of the desired action for the outputs. Output image register locations are designated with a Y followed by a memory location. The actual outputs are updated during the write outputs segment of the scan cycle. There are immediate output instructions available that will update the output points immediately instead of waiting until the write output segment. A complete list of the Immediate instructions is provided in Chapter 5.

The internal control relays (C), the stages (S), and the variable memory (V) are also updated in this segment.

You may recall that you can force various types of points in the system. (This was discussed earlier in this chapter.) If any I/O points or memory data have been forced, the output image register also contains this information.



Normal Run mode scan



Write Outputs

Once the application program has solved the instruction logic and constructed the output image register, the CPU writes the contents of the output image register to the corresponding output points. Remember, the CPU also made sure that any forcing operation changes were stored in the output image register, so the forced points get updated with the status specified earlier.

Diagnostics

During this part of the scan, the CPU performs all system diagnostics and other tasks such as calculating the scan time and resetting the watchdog timer. There are many different error conditions that are automatically detected and reported by the DL105 PLCs. Appendix B contains a listing of the various error codes.

Probably one of the more important things that occurs during this segment is the scan time calculation and watchdog timer control. The DL105 CPU has a "watchdog" timer that stores the maximum time allowed for the CPU to complete the solve application segment of the scan cycle. If this time is exceeded the CPU will enter the Program Mode and turn off all outputs. The default value set from the factory is 200 ms. An error is automatically reported. For example, the Handheld Programmer would display the following message "E003 S/W TIMEOUT" when the scan overrun occurs.

You can use AUX 53 to view the minimum, maximum, and current scan time. Use AUX 55 to increase or decrease the watchdog timer value.

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 that 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 determine the amount of time spent during the various segments of operation.

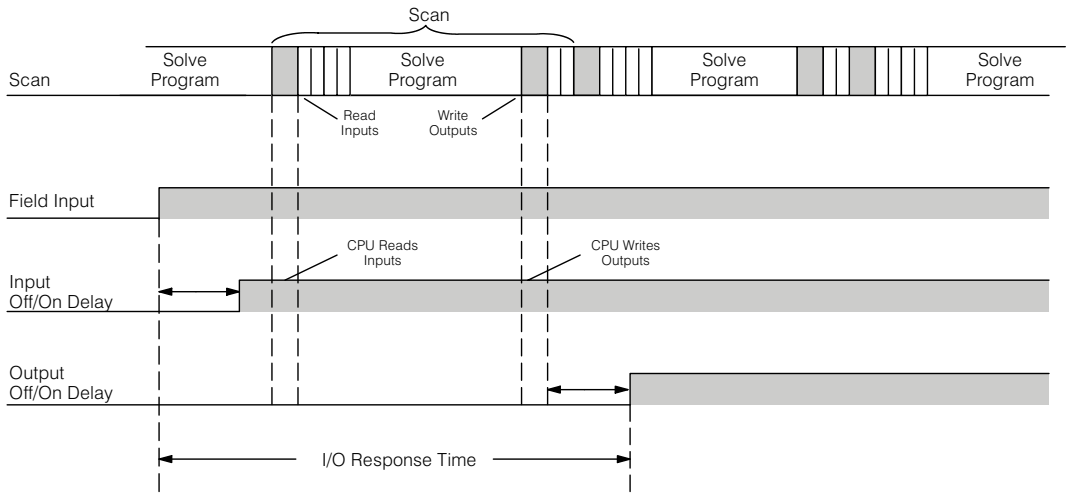
There are four things that can affect the I/O response time.

- The point in the scan cycle when the field input changes states
- Input Off to On delay time
- CPU scan time
- Output 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 input changes just before the Read Inputs portion of the execution cycle. In this case the input status is read, the application program is solved, and the output point gets updated. The following diagram shows an example of the timing for this situation.

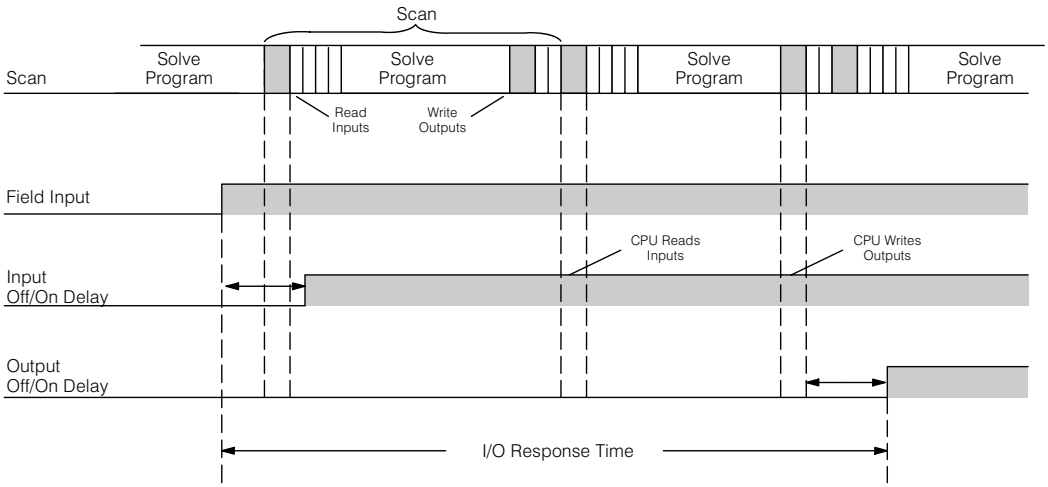


In this case, you can calculate the response time by simply adding the following items:

Input Delay + Scan Time + Output Delay = Response Time

Normal Maximum I/O Response

The I/O response time is longest when the input changes just after the Read Inputs 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:

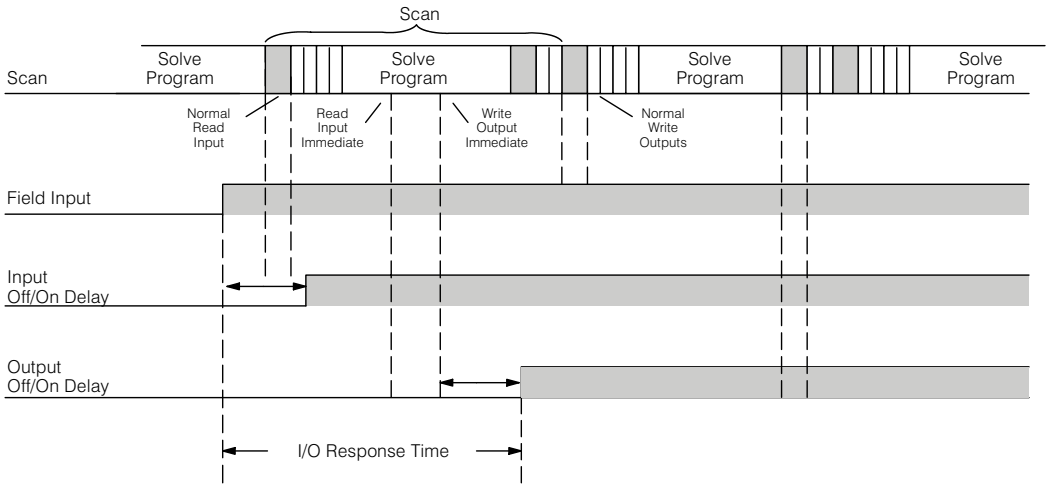
Input Delay +(2 x Scan Time) + Output Delay = Response Time

Improving Response Time

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

- You can choose instructions with faster execution times
- You can use immediate I/O instructions (which update the I/O points during the program execution)
- You can use the HSIO Mode 50 Pulse Catch features designed to operate in high-speed environments. See the Chapter 3 for details on using this feature.

Of these three things the Immediate I/O instructions are probably the most important and most useful. The following example shows how an immediate input instruction and immediate output instruction would affect the response time.



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

Input Delay + Instruction Execution Time + Output Delay = Response Time

The instruction execution time would be calculated by adding the time for the immediate input instruction, the immediate output instruction, and any other instructions in between the two.

NOTE: *Even though the immediate instruction reads the most current status from I/O, it only uses the results to solve that one instruction. It does not use the new status to update the image register. Therefore, any regular instructions that follow will still use the image register values. Any immediate instructions that follow will access the I/O again to update the status.*

CPU Scan Time Considerations

The scan time covers all the cyclical tasks that are performed by the operating system. You can use ***Direct***SOFT or the Handheld Programmer to display the minimum, maximum, and current scan times that have occurred since the previous Program Mode to Run Mode transition. This information can be very important when evaluating the performance of a system.

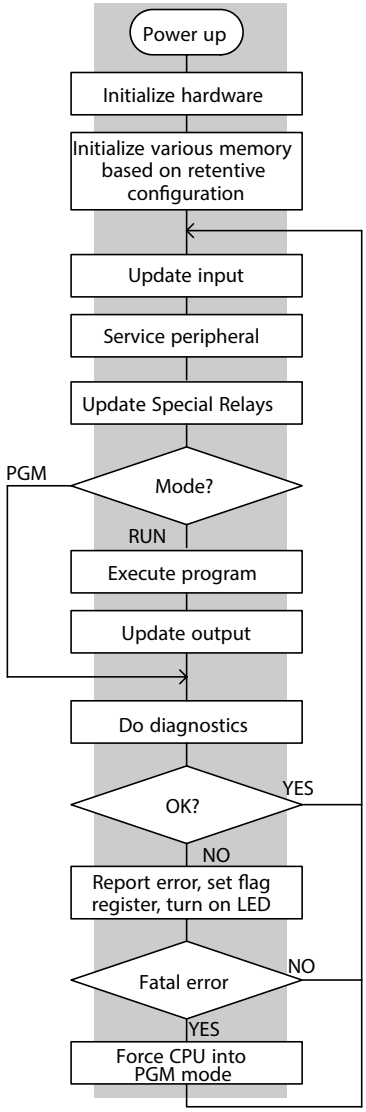
As we've shown previously there are several segments that make up the scan cycle. Each of these segments requires a certain amount of time to complete. Of all the segments, the following are the most important.

- Input Update
- Peripheral Service
- Program Execution
- Output Update
- Timed Interrupt Execution

The only one you really have the most control over is the amount of time it takes to execute the application program. This is because different instructions take different amounts of time to execute. So, if you think you need a faster scan, then you can try to choose faster instructions.

Your choice of I/O type and peripheral devices can also affect the scan time. However, these things are usually dictated by the application.

The following paragraphs provide some general information on how much time some of the segments can require.



Reading Inputs

The time required during each scan to read the input status is 40µS. Don't confuse this with the I/O response time that was discussed earlier.

Writing Outputs

The time required to write the output status is 629µs. Don't confuse this with the I/O response time that was discussed earlier.

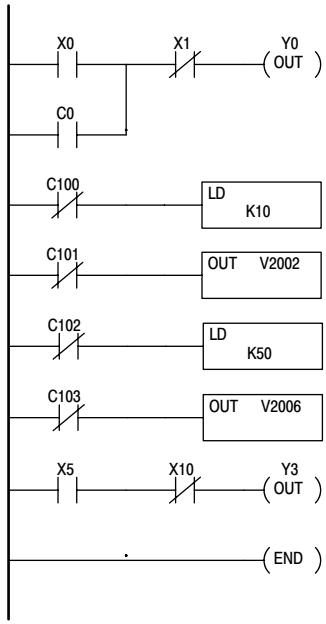
Application Program Execution

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, and the amount of execution overhead.

Just add the execution times for all the instructions in your program to determine to total execution time. Appendix C provides a complete list of the instruction execution times for the DL105 Micro PLC. For example, the execution time for running the program shown below is calculated as follows:

Instruction	Time
STR X0	3.3 µs
OR C0	2.7 µs
ANDN X1	2.7 µs
OUT Y0	3.4 µs
STRN C100	3.9 µs
LD K10	62 µs
STRN C101	3.9 µs
OUT V2002	60 µs
STRN C102	3.9 µs
LD K50	62 µs
STRN C103	3.9 µs
OUT V2006	60 µs
STR X5	3.3 µs
ANDN X10	2.7 µs
OUT Y3	3.4 µs
END	27 µs
TOTAL	308.1 µs

Overhead	DL105
Minimum	0.86 ms
Maximum	3.85 ms



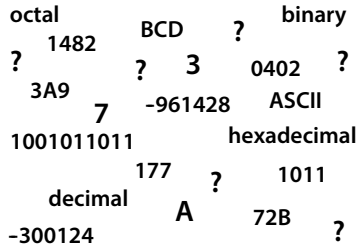
The program above takes only 308.1 µs to execute during each scan. The total scan time is the sum of the program execution plus the overhead as shown above. "Overhead" includes all other housekeeping and diagnostic tasks. The scan time will vary slightly from one scan to the next, because of fluctuation in overhead tasks.

NOTE: *You can move words to EEPROM from within the application program. This can add up to 10ms per 32 word boundary.*

Program Control Instructions — the DL105 PLCs have an interrupt routine feature that changes the way a program executes. Since this instruction interrupts normal program flow, it will have an effect on the program execution time. For example, a timed interrupt routine with a 10ms period interrupts the main program execution (before the END statement) every 10ms, so the CPU can execute the interrupt routine. Chapter 5 provides detailed information on interrupts.

PLC Numbering Systems

If you are a new PLC user or are using ***Direct***LOGIC PLCs for the first time, please take a moment to study how our PLCs use numbers. You'll find that each PLC manufacturer has their own conventions on the use of numbers in their PLCs. We want to take just a moment to familiarize you with how numbers are used in ***Direct***LOGIC PLCs. The information you learn here applies to all our PLCs!

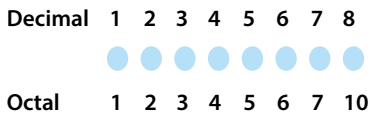


As any good computer does, PLCs store and manipulate numbers in binary form: just ones and zeros. So why do we have to deal with numbers in so many different forms? Numbers have meaning, and some representations are more convenient than others for particular purposes. Sometimes we use numbers to represent a size or amount of something. Other numbers refer to locations or addresses, or to time. In science we attach engineering units to numbers to give a particular meaning (see Appendix G for numbering system details).

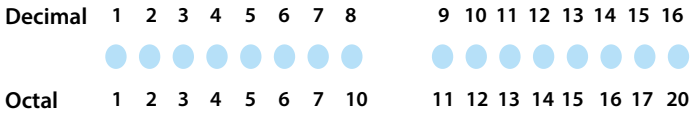
PLC Resources

PLCs offer a fixed amount of resources, depending on the model and configuration. We use the word "resources" to include variable memory (V-memory), I/O points, timers, counters, etc. Most modular PLCs allow you to add I/O points in groups of eight. In fact, all the resources of our PLCs are counted in octal. It's easier for computers to count in groups of eight than ten, because eight is an even power of 2.

Octal means simply counting in groups of eight things at a time. In the figure to the right, there are eight circles. The quantity in decimal is "8", but in octal it is "10" (8 and 9 are not valid in octal). In octal, "10" means 1 group of 8 plus 0 (no individuals).

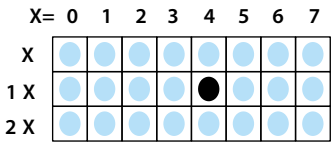


In the figure below, we have two groups of eight circles. Counting in octal we have "20" items, meaning 2 groups of eight, plus 0 individuals Don't say "twenty", say "two- zero octal". This makes a clear distinction between number systems.



After counting PLC resources, it's time to access PLC resources (there's a difference). The CPU instruction set accesses resources of the PLC using octal addresses. Octal addresses are the same as octal quantities, except they start counting at zero. The number zero is significant to a computer, so we don't skip it.

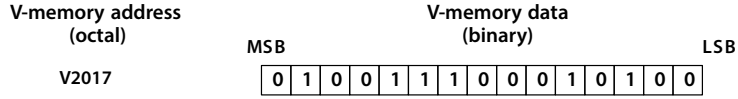
Our circles are in an array of square containers to the right. To access a resource, our PLC instruction will address its location using the octal references shown. If these were counters, "CT14" would access the black circle location.



V- Memory

Variable memory (called "V-memory") stores data for the ladder program and for configuration settings (see Appendix E). V-memory locations and V-memory addresses are the same thing, and are numbered in octal. For example, V2073 is a valid location, while V1983 is not valid ("9" and "8" are not valid octal digits).

Each V-memory location is one data word wide, meaning 16 bits. For configuration registers, our manuals will show each bit of a V-memory word. The least significant bit (LSB) will be on the right, and the most significant bit (MSB) on the left. We use the word "significant" referring to the relative binary weighting of the bits.

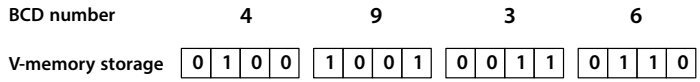


V-memory data is 16-bit binary, but we rarely program the data registers one bit at a time. We use instructions or viewing tools that let us work with decimal, octal, and hexadecimal numbers. All these are converted and stored as binary for us.

A frequently-asked question is "How do I tell if a number is octal, BCD, or hex"? The answer is that we usually cannot tell just by looking at the data... but it does not really matter. What matters is: the source or mechanism which writes data into a V-memory location and the thing which later reads it must both use the same data type (i.e., octal, hex, binary, or whatever). The V-memory location is just a storage box... that's all. It does not convert or move the data on its own.

Binary-Coded Decimal Numbers

Since humans naturally count in decimal (10 fingers, 10 toes), we prefer to enter and view PLC data in decimal as well. However, computers are more efficient in using pure binary numbers. A compromise solution between the two is Binary-Coded Decimal (BCD) representation. A BCD digit ranges from 0 to 9, and is stored as four binary bits (a nibble). This permits each V-memory location to store four BCD digits, with a range of decimal numbers from 0000 to 9999.



In a pure binary sense, a 16-bit word can represent numbers from 0 to 65535. In storing BCD numbers, the range is reduced to only 0 to 9999. Many math instructions use Binary-Coded Decimal (BCD) data, and ***Direct***SOFT and the handheld programmer allow us to enter and view data in BCD.

Hexadecimal Numbers

Hexadecimal numbers are similar to BCD numbers, except they utilize all possible binary values in each 4-bit digit. They are base-16 numbers so we need 16 different digits. To extend our decimal digits 0 through 9, we use A through F as shown.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

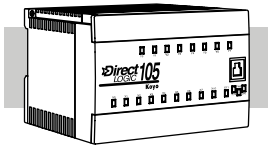
A 4-digit hexadecimal number can represent all 65536 values in a V-memory word. The range is from 0000 to FFFF (hex). PLCs often need this full range for sensor data, etc. Hexadecimal is just a convenient way for humans to view full binary data.

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 DL105 Micro PLCs. A memory map overview for the CPU follows the memory descriptions.

Octal Numbering System

All memory locations and resources 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.

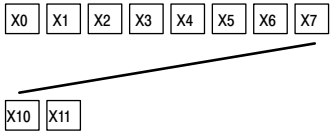


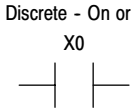
Discrete and Word Locations

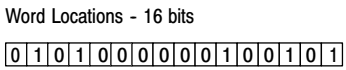
As you examine the different memory types, you'll notice two types of memory in the DL105, discrete and word memory. Discrete memory is one bit that can be either a 1 or a 0. Word memory is referred to as V-memory (variable) and is a 16-bit location normally used to manipulate data/numbers, store data/numbers, etc.

Some information is automatically stored in V-memory. For example, the timer current values are stored in V-memory.

As you examine the different memory types, you'll notice two types of memory in the DL105, discrete and word memory. Discrete memory is one bit that can be either a 1 or a 0. Word memory is referred to as V-memory (variable) and is a 16-bit location normally used to manipulate data/numbers, store data/numbers, etc.



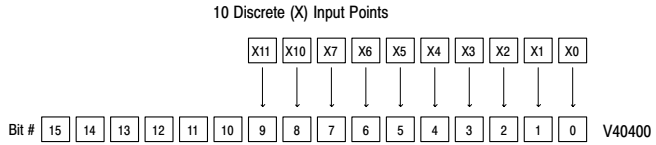




Some information is automatically stored in V-memory. For example, the timer current values are stored in V-memory.

V-memory Locations for Discrete Memory Areas

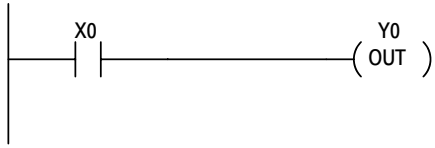
The discrete memory area is for inputs, outputs, control relays, special relays, stages, timer status bits and counter status bits. However, you can also access the bit data types as a V-memory word. Each V-memory location contains 16 consecutive discrete locations. For example, the following diagram shows how the X input points are mapped into V-memory locations.



These discrete memory areas and their corresponding V-memory ranges are listed in the memory area table for DL105 Micro PLCs on the following pages.

Input Points (X Data Type)

The discrete input points are noted by an X data type. There are 10 discrete input points available with DL105 CPUs. In this example, the output point Y0 will be turned on when input X0 energizes.



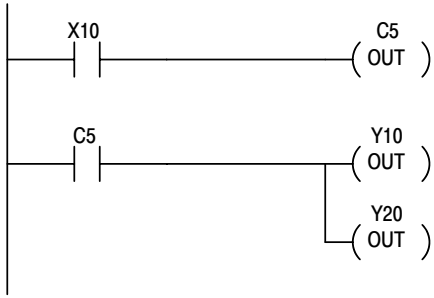
Output Points (Y Data Type)

The discrete output points are noted by a Y data type. There are 128 discrete output points available with DL105 CPUs. In this example, output point Y1 will be turned on when input X1 energizes.

Control Relays (C Data Type)

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 16 consecutive discrete locations.

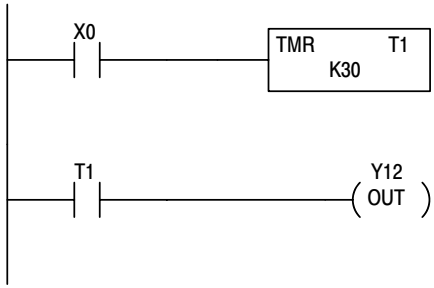
In this example, memory location C5 will energize when input X10 turns on. The second rung shows a simple example of how to use a control relay as an input.



Timers and Timer Status Bits (T Data type)

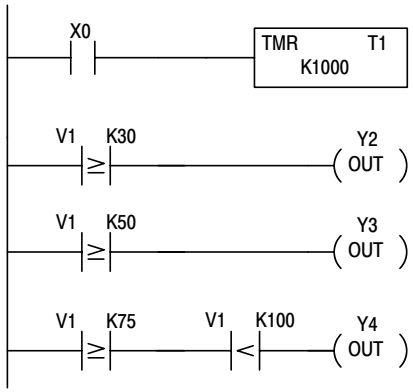
Timer status bits reflect the relationship between the current value and the preset value of a specified timer. The timer status bit will be on when the current value is equal or greater than the preset value of a corresponding timer.

When input X0 turns on, timer T1 will start. When the timer reaches the preset of 3 seconds (K of 30) timer status contact T1 turns on. When T1 turns on, output Y12 turns on. Turning off X0 resets the timer.



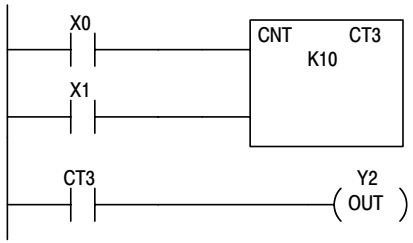
Timer Current Values (V Data Type)

As mentioned earlier, some information is automatically stored in V-memory. This is true for the current values associated with timers. For example, V0 holds the current value for Timer 0, V1 holds the current value for Timer 1, etc.



Counters and Counter Status Bits (CT Data type)

The primary reason for his is programming flexibility. The example shows how you can use relational contacts to monitor several time intervals from a single timer.

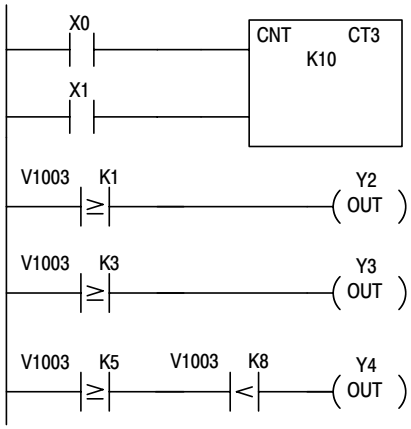


Each time contact X0 transitions from off to on, the counter increments by one. (If X1 comes on, the counter is reset to zero.) When the counter reaches the preset of 10 counts (K of 10) counter status contact CT3 turns on. When CT3 turns on, output Y12 turns on.

Counter Current Values (V Data Type)

Just like the timers, the counter current values are also automatically stored in V-memory. For example, V1000 holds the current value for Counter CT0, V1001 holds the current value for Counter CT1, etc.

The primary reason for his is programming flexibility. The example shows how you can use relational contacts to monitor the counter values.

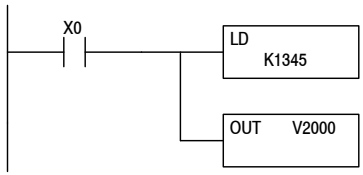


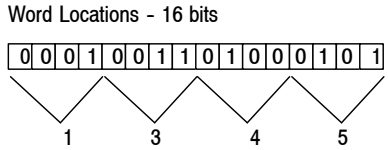
Word Memory (V Data Type)

Word memory is referred to as V-memory (variable) and is a 16-bit location normally used to manipulate data/numbers, store data/numbers, etc. (see Appendix E).

Some information is automatically stored in V-memory. For example, the timer current values are stored in V-memory.

The example shows how a four-digit BCD constant is loaded into the accumulator and then stored in a V-memory location.

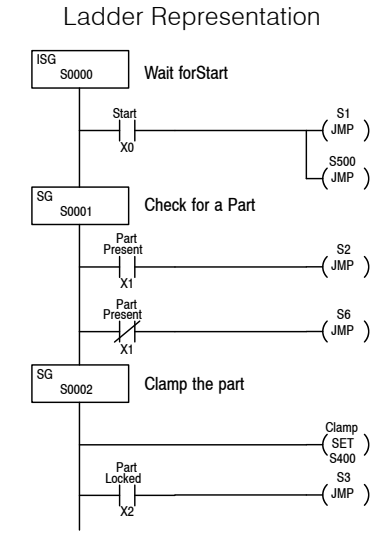




Stages (S Data type)

Stages are used in RLL^{PLUS} Stage 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 7 for a more detailed description of RLL^{PLUS} Stage 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.



Special Relays (SP Data Type)

Special relays are discrete memory locations with pre-defined functionality. There are many different types of special relays. For example, some aid in program development, others provide system operating status information, etc. Appendix D provides a complete listing of the special relays.

In this example, control relay C10 will energize for 50ms and de-energize for 50ms because SP5 is a pre-defined relay that will be on for 50ms and off for 50ms.

SP4: 1 second clock
SP5: 100 ms clock
SP6: 50 ms clock

DL105 System V-memory

System Parameters and Default Data Locations (V Data Type)

The DL105 PLCs reserve several V-memory locations for storing system parameters or certain types of system data. These memory locations store things like the error codes, High-Speed I/O data, and other types of system setup information.

System V-memory	Description of Contents	Default Values/ Ranges
V2320–V2377	The default location for multiple preset values for the High-Speed Counter	N/A
V7620–V7627	Locations for DV-1000 operator interface parameters	
V7620	Sets the V-memory location that contains the value.	V0–V2377
V7621	Sets the V-memory location that contains the message.	V0–V2377
V7622	Sets the total number (1–16) of V-memory locations to be displayed.	1–16
V7623	Sets the V-memory location that contains the numbers to be displayed.	V0–V2377
V7624	Sets the V-memory location that contains the character code to be displayed.	V0–V2377
V7625	Contains the function number that can be assigned to each key.	V-memory location for X, Y, or C points used.
V7626	Powerup operational mode.	0, 1, 2, 12, 3
V7627	Change preset value.	0000 to 9999
V7630	Starting location for the multi- step presets for channel 1. The default value is 2320, which indicates the first value should be obtained from V2320. Since there are 24 presets available, the default range is V2320–V2377. You can change the starting point if necessary.	Default: V2320 Range: V0–V2320
V7631–V7632	Not used	N/A
V7633	Sets the desired function code for the high speed counter, interrupt, pulse catch, pulse train, and input filter. Location is also used for setting the power-up in Run Mode option.	Default: 0060 Lower Byte Range: 10 - Counter 20 - Quadrature 30 - Pulse Out 40 - Interrupt 50 - Pulse Catch 60 - Filtered discrete in Upper Byte Range: Bits 8–12, 14, 15: Unused Bit 13: Power-up in Run"
V7634	X0 Setup Register for High-Speed I/O functions	Default: 1006
V7635	X1 Setup Register for High-Speed I/O functions	Default: 1006
V7636	X2 Setup Register for High-Speed I/O functions	Default: 1006
V7637	X3 Setup Register for High-Speed I/O functions	Default: 1006

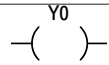
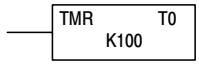
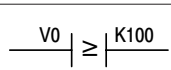
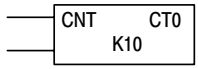
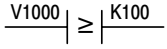
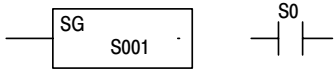
System V-memory	Description of Contents	Default Values/ Ranges
V7640–V7642	Locations for DV-1000 operator interface parameters.	
V7640	Title Timer preset value pointer.	V2000–V2377
V7641	Titled Counter preset value pointer.	V2000–V2377
V7642	HiByte- Titled Timer preset block size, LoByte- Titled counter preset block size."	1–99
V7643–V7647	Not used	
V7751	Fault Message Error Code — stores the 4-digit code used with the FAULT instruction when the instruction is executed.	N/A
V7752–V7754	Not used	N/A
V7755	Error code — stores the fatal error code.	
V7756	Error code — stores the major error code.	
V7757	Error code — stores the minor error code.	
V7760–V7762	Not used	
V7763	Program address where syntax error exists	N/A
V7764	Syntax error code	N/A
V7765	Scan — stores the total number of scan cycles that have occurred since the last Program Mode to Run Mode transition.	N/A
V7666–V7774	Not used	N/A
V7775	Scan — stores the current scan time (milliseconds).	N/A
V7776	Scan — stores the minimum scan time that has occurred since the last Program Mode to Run Mode transition (milliseconds).	N/A
V7777	Scan — stores the maximum scan time that has occurred since the last Program Mode to Run Mode transition (milliseconds).	N/A

DL105 Aliases

An alias is an alternate way of referring to certain memory types, such as timer/counter current values, V-memory locations for I/O points, etc., which simplifies understanding the memory address. 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 to reference memory locations.

Address Start	Alias Start	Example
V0	TA0	V0 is the timer accumulator value for timer 0, therefore, it's alias is TA0. TA1 is the alias for V1, etc..
V1000	CTA0	V1000 is the counter accumulator value for counter 0, therefore, it's alias is CTA0. CTA1 is the alias for V1001, etc.
V40000	VGX0	V40000 is the word memory reference for discrete bits GX0 through GX17, therefore, it's alias is VGX0. V40001 is the word memory reference for discrete bits GX20 through GX37, therefore, it's alias is VGX20.
V40200	VGY0	V40200 is the word memory reference for discrete bits GY0 through GY17, therefore, it's alias is VGY0. V40201 is the word memory reference for discrete bits GY20 through GY37, therefore, it's alias is VGY20.
V40400	VX0	V40400 is the word memory reference for discrete bits X0 through X17, therefore, it's alias is VX0. V40401 is the word memory reference for discrete bits X20 through X37, therefore, it's alias is VX20.
V40500	VY0	V40500 is the word memory reference for discrete bits Y0 through Y17, therefore, it's alias is VY0. V40501 is the word memory reference for discrete bits Y20 through Y37, therefore, it's alias is VY20.
V40600	VC0	V40600 is the word memory reference for discrete bits C0 through C17, therefore, it's alias is VC0. V40601 is the word memory reference for discrete bits C20 through C37, therefore, it's alias is VC20.
V41000	VS0	V41000 is the word memory reference for discrete bits S0 through S17, therefore, it's alias is VS0. V41001 is the word memory reference for discrete bits S20 through S37, therefore, it's alias is VS20.
V41100	VT0	V41100 is the word memory reference for discrete bits T0 through T17, therefore, it's alias is VT0. V41101 is the word memory reference for discrete bits T20 through T37, therefore, it's alias is VT20.
V41140	VCT0	V41140 is the word memory reference for discrete bits CT0 through CT17, therefore, it's alias is VCT0. V41141 is the word memory reference for discrete bits CT20 through CT37, therefore, it's alias is VCT20.
V41200	VSP0	V41200 is the word memory reference for discrete bits SP0 through SP17, therefore, it's alias is VSP0. V41201 is the word memory reference for discrete bits SP20 through SP37, therefore, it's alias is VSP20.

DL105 Memory Map

Memory Type	Discrete Memory Reference (octal)	Word Memory Reference (octal)	Qty Decimal	Symbol
Input Points (See note 1)	X0–X177	V40400–V40407	128	
Output Points (See note 1)	Y0–Y177	V40500–V40507	128	
Control Relays	C0–C377	V40600–V40617	256	
Special Relays	SP0–SP117 SP540–SP577	V41200–V41204 V41226–V41227	112	
Timers	T0–T77		64	
Timer Current Values	None	V0–V77	64	
Timer Status Bits	T0–T77	V41100–V41103	64	
Counters	CT0–CT77		64	
Counter Current Values	None	V1000–V1077	64	
Counter Status Bits	CT0–CT77	V41140–V41143	64	
Data Words (See Appendix E)	None	V2000–V2377	256	None specific, used with many instructions
Data Words Non–Volatile (See Appendix E)	None	V4000–V4177	128	None specific, used with many instructions
Stages	S0–S377	V41000–V41017	256	
System parameters	None	V7620–V7647 V7750–V7777	48	None specific, used for various purposes

1. The DL105 systems are limited to 10 discrete inputs and 8 discrete outputs with the present available hardware however, 128 point addresses exist.

X Input Bit Map

This table provides a listing of individual Input points associated with each V-memory address bit for the DL105's ten physical inputs. Actual available references are X0 to X177 (V40400 –V40407).

MSB	DL105 Output (Y) Points														LSB	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
-	-	-	-	-	-	011	010	007	006	005	004	003	002	001	000	V40400

Y Output Bit Map

This table provides a listing of individual output points associated with each V-memory address bit for the DL105's eight physical outputs. Actual available references are Y0 to Y177 (V40500 –V40507).

MSB	DL105 Output (Y) Points														LSB	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
-	-	-	-	-	-	-	-	007	006	005	004	003	002	001	000	V40500

Control Relay Bit Map

This table provides a listing of the individual control relays associated with each V-memory address bit.

MSB	DL105 Control Relays (C)														LSB	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
017	016	015	014	013	012	011	010	007	006	005	004	003	002	001	000	V40600
037	036	035	034	033	032	031	030	027	026	025	024	023	022	021	020	V40601
057	056	055	054	053	052	051	050	047	046	045	044	043	042	041	040	V40602
077	076	075	074	073	072	071	070	067	066	065	064	063	062	061	060	V40603
117	116	115	114	113	112	111	110	107	106	105	104	103	102	101	100	V40604
137	136	135	134	133	132	131	130	127	126	125	124	123	122	121	120	V40605
157	156	155	154	153	152	151	150	147	146	145	144	143	142	141	140	V40606
177	176	175	174	173	172	171	170	167	166	165	164	163	162	161	160	V40607
217	216	215	214	213	212	211	210	207	206	205	204	203	202	201	200	V40610
237	236	235	234	233	232	231	230	227	226	225	224	223	222	221	220	V40611
257	256	255	254	253	252	251	250	247	246	245	244	243	242	241	240	V40612
277	276	275	274	273	272	271	270	267	266	265	264	263	262	261	260	V40613
317	316	315	314	313	312	311	310	307	306	305	304	303	302	301	300	V40614
337	336	335	334	333	332	331	330	327	326	325	324	323	322	321	320	V40615
357	356	355	354	353	352	351	350	347	346	345	344	343	342	341	340	V40616
377	376	375	374	373	372	371	370	367	366	365	364	363	362	361	360	V40617

Stage Control / Status Bit Map

This table provides a listing of individual Stage control bits associated with each V-memory address bit.

MSB	DL105 Stage (S) Control Bits														LSB	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
017	016	015	014	013	012	011	010	007	006	005	004	003	002	001	000	V41000
037	036	035	034	033	032	031	030	027	026	025	024	023	022	021	020	V41001
057	056	055	054	053	052	051	050	047	046	045	044	043	042	041	040	V41002
077	076	075	074	073	072	071	070	067	066	065	064	063	062	061	060	V41003
117	116	115	114	113	112	111	110	107	106	105	104	103	102	101	100	V41004
137	136	135	134	133	132	131	130	127	126	125	124	123	122	121	120	V41005
157	156	155	154	153	152	151	150	147	146	145	144	143	142	141	140	V41006
177	176	175	174	173	172	171	170	167	166	165	164	163	162	161	160	V41007
217	216	215	214	213	212	211	210	207	206	205	204	203	202	201	200	V41010
237	236	235	234	233	232	231	230	227	226	225	224	223	222	221	220	V41011
257	256	255	254	253	252	251	250	247	246	245	244	243	242	241	240	V41012
277	276	275	274	273	272	271	270	267	266	265	264	263	262	261	260	V41013
317	316	315	314	313	312	311	310	307	306	305	304	303	302	301	300	V41014
337	336	335	334	333	332	331	330	327	326	325	324	323	322	321	320	V41015
357	356	355	354	353	352	351	350	347	346	345	344	343	342	341	340	V41016
377	376	375	374	373	372	371	370	367	366	365	364	363	362	361	360	V41017

Timer Status Bit Map

This table provides a listing of individual timer contacts associated with each V-memory address bit.

MSB	DL105 Timer (T) Contacts														LSB	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
017	016	015	014	013	012	011	010	007	006	005	004	003	002	001	000	V41100
037	036	035	034	033	032	031	030	027	026	025	024	023	022	021	020	V41101
057	056	055	054	053	052	051	050	047	046	045	044	043	042	041	040	V41102
077	076	075	074	073	072	071	070	067	066	065	064	063	062	061	060	V41103

Counter Status Bit Map

This table provides a listing of individual counter contacts associated with each V-memory address bit.

MSB	DL105 Counter (CT) Contacts														LSB	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
017	016	015	014	013	012	011	010	007	006	005	004	003	002	001	000	V41140
037	036	035	034	033	032	031	030	027	026	025	024	023	022	021	020	V41141
057	056	055	054	053	052	051	050	047	046	045	044	043	042	041	040	V41142
077	076	075	074	073	072	071	070	067	066	065	064	063	062	061	060	V41143