14. PLC MEMORY

Topics:

- PLC-5 memory types; program and data
- Data types; output, input, status, bit, timer, counter, integer, floating point, etc.
- Memory addresses; words, bits, data files, expressions, literal values and indirect.

Objectives:

- To know the basic memory types available
- To be able to use addresses for locations in memory

14.1 INTRODUCTION

Advanced ladder logic functions allow controllers to perform calculations, make decisions and do other complex tasks. Timers and counters are examples of ladder logic functions. They are more complex than basic input contacts and output coils and they rely upon data stored in the memory of the PLC. The memory of the PLC is organized to hold different types of programs and data.

14.2 MEMORY ADDRESSES

The memory in a PLC is organized by data type as shown in Figure 14.1. There are two fundamental types of memory used in Allen-Bradley PLCs - Program and Data memory. Memory is organized into blocks of up to 1000 elements in an array called a file. The Program file holds programs, such as ladder logic. There are eight Data files defined by default, but additional data files can be added if they are needed.

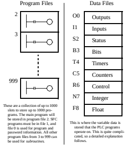


Figure 14.1 PLC Memory

14.3 PROGRAM FILES

In a PLC-5 the first three program files, from 0 to 2, are defined by default. File 0 contains system information and should not be changed, and file 1 is reserved for SFCs. File 2 is available for user programs and the PLC will run the program in file 2 by default. Other program files can be added from file 3 to 999. Typical reasons for creating other programs are for subroutines.

When a user creates a ladder logic program with programming software, it is converted to a mnemonic-like form, and then transferred to the PLC, where it is stored in a program file. The contents of the program memory cannot be changed while the PLC is running. If, while a program was running, it was overwritten with a new program, serious problems could arise.

14.4 DATA FILES

Data files are used for storing different information types, as shown in Figure 14.2. These locations are numbered from 0 to 999. The letter in front of the number indicates the data type. For example, *F8:* is read as *floating point numbers* in *data file 8*. Numbers are not given for *O:* and *I:*, but they are implied to be *O0:* and *I1:*. The number that follows the *:* is the location number. Each file may contain from 0 to 999 locations that may store values. For the input *I:* and output *O:* files the locations are converted to physical locations on the PLC using rack and slot numbers. The addresses that can be used will depend upon the hardware configuration. The status *S2:* file is more complex and is discussed later. The other memory locations are simply slots to store data in. For example, *F8:35* would indicate the 36th value in the 8th data file which is floating point numbers.

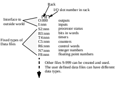


Figure 14.2 Data Files for an Allen Bradley PLC-5

Only the first three data files are fixed *O:*, *I:* and *S2:*, all of the other data files can be moved. It is also reasonable to have multiple data files with the same data type. For example, there could be two files for integer numbers *N7:* and *N10:*. The length of the data files can be from 0 up to 999 as shown in Figure 14.3. But, these files are often made smaller to save memory.

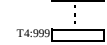


Figure 14.3 Locations in a Data File

Figure 14.2 shows the default data files for a PLC-5. There are many additional data types, a full list is shown in Figure 14.4. Some of the data types are complex and contain multiple data values, including *BT, C, MG, PD, R, SC*, and *T*. Recall that timers require integers for the accumulator and preset, and TT, DN and EN bits are required. Other data types are based on single bits, 8 bit bytes and 16 bit words.

Type	Length (words)
A - ASCII	1/2
B - bit	1/16
BT - block transfer	6
C - counter	3
D - BCD	1
F - floating point	2
MG - message	56
N - integer (signed, unsigned, 2s compliment, BCD)	1
PD - PID controller	82
R - control	3
SC - SFC status	3
ST - ASCII string	42
T - timer	3

NOTE: Memory is a general term that refers to both files and locations. The term *file* is specific to PLC manufacturers and is not widely recognized elsewhere.

Figure 14.4 Allen-Bradley Data Types

When using data files and functions we need to ask for information with an address. The simplest data addresses are data bits (we have used these for basic inputs and outputs already). An example of Address bits is shown in Figure 14.5. Memory bits are normally indicated with a forward slash followed by a bit number */n*. The first example is from an input card *I:000*, the third input is indicated with the bit address */02*. The second example is for a counter *C5:* done bit */DN*. This could also be replaced with *C5:4/15* to get equivalent results. The *DN* notation, and others like it are used to simplify the task of programming. The example *B3/4* will get the fourth bit in bit memory *B3*. For bit memory the slash is not needed, because the data type is already bits.

bit - individual bits in accessed - this is like addressing a single output as a data bit.

I:000/02 - the third input bit from input card I:000
C5:4/DN - the DN bit of a counter
B3/4 - the fourth bit in bit memory

NOTE: Some bit addresses, especially inputs and outputs are addressed using octal. This often leads to careless errors and mistakes. For example if you want the 11th output bit, or bit 10, you would need to use 12 in octal to address it properly.

1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th	15th	16th
00	01	02	03	04	05	06	07	10	11	12	13	14	15	16	17

Figure 14.5 Bit Level Addressing

Entire words can be addressed as shown in Figure 14.6. These values will normally be assumed to be 2s compliment, but some functions may assume otherwise. The first example shows a simple integer memory value. The next example gets up to inputs (from card 0 in rack zero) as a single word. The last two examples are more complex and they access the accumulator and preset values for a timer. Here a '.' is used as the '/' was used for bit memory to indicate it is an integer. The first two examples don't need the '.' because they are both integer value types. Other types of word addressing are possible, including floating point numbers.

integer word - 16 bits can be manipulated as an integer.

N7:8 - the 9th value from integer memory
I:000 - an integer with all input values from an input card
T4:7.ACC - the accumulator value for a timer
T4:7.PRE - the preset value for a timer

Figure 14.6 Integer Word Addressing

Data values do not always need to be stored in memory, they can be define literally. Figure 14.7 shows an example of two different data values. The first is an integer, the second is a real number. Hexadecimal numbers can be indicated by following the number with *H*, a leading zero is also needed when the first digit is *A, B, C, D, E* or *F*. A binary number is indicated by adding a *B* to the end of the number.

literal data value - a data value can be provided without storing it in memory.

8 - an integer
8.5 - a floating point number
08FH - a hexadecimal value *8F*
01101101B - a binary number *01101101*

Figure 14.7 Literal Data Values

Sometimes we will want to refer to an array of values, as shown in Figure 14.8. This data type is indicated by beginning the number with a pound or hash sign '#'. The first example describes an array of floating point numbers staring in file *8* at location *5*. The second example is for an array of integers in file *7* starting at location *0*. The length of the array is determined elsewhere.

file - the first location of an array of data values.

#F8:5 - indicates a group of values starting at F8:5
#N7:0 - indicates a group of values starting at N7:0

Figure 14.8 File Addressing

Indirect addressing is a method for allowing a variable in a data address, as shown in Figure 14.9. The indirect (variable) part of the address is shown between brackets '[' and ']'. If a PLC is looking at an address and it finds an indirect address it will look in the specified memory location, and put that number in place of the indirect address. Consider the first example below *I:000/[N7:2]*, if the value in the integer memory location *N7:2* is 45, then the address becomes *I:000/45*. The other examples are very similar. This type of technique is very useful when making programs that can be adapted for different recipes - by changing a data value in one memory location the program can follow a new set of data.

indirect - another memory location can be used in the description of a location.

I:000/[N7:2] -If N7:2 location contains 5 this will become I:000/05
I:[N7:1]/03 -If the integer memory location contains 2 this will become I:002/03
#I:[N7:1] -If the integer memory location contains 2 the file will start at I:002
N[N7:0]:8 - If the number in N7:0 is 10 the data address becomes N10:8

Figure 14.9 Indirect Addressing

Expressions allow addresses and functions to be typed in and interpreted when the program is run. The example in Figure 14.10 will get a floating point number from file 8, location 3, perform a sine transformation, and then add 1.3. The text string is not interpreted until the PLC is running, and if there is an error, it may not occur until the program is running - so use this function cautiously.

expression - a text string that describes a complex operation.

"sin(F8:3) + 1.3" - a simple calculation

Figure 14.10 Expression Data Values

These data types and addressing modes will be discussed more as applicable functions are presented later in this chapter and book. Floating point numbers, expressions and indirect addressing may not be available on older or lower cost PLCs.

Figure 14.11 shows a simple example ladder logic with functions. The basic operation is such that while input *A* is true the functions will be performed. The first statement will move (MOV) the literal value of *130* into integer memory *N7:0*. The next move function will copy the value from *N7:0* to *N7:1*. The third statement will add integers value in *N7:0* and *N7:1* and store the results in *N7:2*.

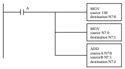
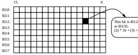


Figure 14.11 An Example of Ladder Logic Functions

14.4.1 User Bit Memory

Individual data bits can be accessed in the bit memory. These can be very useful when keeping track of internal states that do not directly relate to an output or input. The bit memory can be accessed with individual bits or with integer words. Examples of bit addresses are shown in Figure 14.12. The single blackened bit is in the third word *B3:2* and it is the 4th bit *03*, so it can be addressed with *B3:2/03*. Overall, it is the 35th bit, so it could also be addressed with *B3/35*.



Other Examples: B3:0/0 = B3/0
B3:0/10 = B3/10
B3:1/0 = B3/16
B3:1/5 = B3/21
B3:2/0 = B3/32
etc...

Figure 14.12 Bit Memory

This method can also be used to access bits in integer memory also.

14.4.2 Timer Counter Memory

Previous chapters have discussed the operation of timers and counters. The ability to address their memory directly allows some powerful tools. Recall that by default timers are stored in the *T4:* file. The bits and words for timers are;

EN - timer enabled bit (bit 15)
TT - timer timing bit (bit 14)
DN - timer done bit (bit 13)
PRE - preset word
ACC - accumulated time word

Counter are stored in the *C5:* file and they have the following bits and words.

CU - count up bit (bit 15)
CD - count down bit (bit 14)
DN - counter done bit (bit 13)
OV - overflow bit (bit 12)
UN - underflow bit (bit 11)
PRE - preset word
ACC - accumulated count word

As discussed before we can access timer and counter bits and words using the proper notation. Examples of these are shown in Figure 14.13. The bit values can only be read, and should not be changed. The presets and accumulators can be read and overwritten.

Words
T4:0.PRE - the preset value for timer T4:0
T4:0.ACC - the accumulated value for timer T4:0
C5:0.PRE - the preset value for counter C5:0
C5:0.ACC - the accumulated value for counter C5:0

Bits
T4:0/EN - indicates when the input to timer T4:0 is true
T4:0/TT - indicates when the timer T4:0 is counting
T4:0/DN - indicates when timer T4:0 has reached the maximum
C5:0/CU - indicates when the count up instruction is true for C5:0
C5:0/CD - indicates when the count down instruction is true for C5:0
C5:0/DN - indicates when the counter C5:0 has reached the preset
C5:0/OV - indicates when the counter C5:0 has reached the maximum value (32767)
C5:0/UN - indicates when the counter C5:0 has reached the minimum value (-32768)

Figure 14.13 Examples of Timer and Counter Addresses

Consider the simple ladder logic example in Figure 14.14. It shows the use of a timer timing *TT* bit to seal on the timer when a door input has gone true. While the timer is counting, the bit will stay true and keep the timer counting. When it reaches the 10 second delay the *TT* bit will turn off. The next line of ladder logic will turn on a light while the timer is counting for the first 10 seconds.

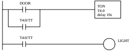


Figure 14.14 Door Light Example

14.4.3 PLC Status Bits (for PLC-5s and Micrologix)

Status memory allows a program to check the PLC operation, and also make some changes. A selected list of status bits is shown in Figure 14.15 for Allen-Bradley Micrologic and PLC-5 PLCs. More complete lists are available in the manuals. For example the first four bits *S2:0/x* indicate the results of calculations, including carry, overflow, zero and negative/sign. The *S2:1/15* will be true once when the PLC is turned on - this is the first scan bit. The time for the last scan will be stored in *S2:8*. The date and clock can be stored and read from locations *S2:18* to *S2:23*.

S2:0/0 carry in math operation
S2:0/1 overflow in math operation
S2:0/2 zero in math operation
S2:0/3 sign in math operation
S2:1/15 first scan of program file
S2:8 the scan time (ms)
S2:18 year
S2:19 month
S2:20 day
S2:21 hour
S2:22 minute
S2:23 second
S2:28 watchdog setpoint
S2:29 fault routine file number
S2:30 STI (selectable timed interrupt) setpoint
S2:31 STI file number
S2:46-S2:54,S2:55-S2:56 PII (Programmable Input Interrupt) settings
S2:55 STI last scan time (ms)
S2:77 communication scan time (ms)

Figure 14.15 Status Bits and Words for Micrologix and PLC-5s

The other status words allow more complex control of the PLC. The watchdog timer allows a time to be set in *S2:28* so that if the PLC scan time is too long the PLC will give a fault condition - this is very important for dangerous processes. When a fault occurs the program number in *S2:29* will run. For example, if you have a divide by zero fault, you can run a program that recovers from the error, if there is no program the PLC will halt. The locations from *S2:30* to *S2:55* are used for interrupts. Interrupts can be used to run programs at fixed time intervals, or when inputs change.

14.4.4 User Function Control Memory

Simple ladder logic functions can complete operations in a single scan of ladder logic. Other functions such as timers and counters will require multiple ladder logic scans to finish. While timers and counters have their own memory for control, a generic type of control memory is defined for other function. This memory contains the bits and words in Figure 14.16. Any given function will only use some of the values. The meaning of particular bits and words will be described later when discussing specific functions.

EN - enable bit (bit 15)
EU - enable unload (bit 14)
DN - done bit (bit 13)
EM - empty bit (bit 12)
ER - error bit (bit 11)
UL - unload bit (bit 10)
IN - inhibit bit (bit 9)
FD - found bit (bit 8)
LEN - length word
POS - position word

Figure 14.16 Bits and Words for Control Memory

14.4.5 Integer Memory

Integer memory is 16 bit words that are normally used as 2s compliment numbers that can store data values from -32768 to +32767. When decimal fractions are supplied they are rounded to the nearest number. These values are normally stored in N7:xx by default, but new blocks of integer memory are often created in other locations such as *N9:xx*. Integer memory can also be used for bits.

14.4.6 Floating Point Memory

Floating point memory is available in newer and higher cost PLCs, it is not available on the Micrologix. This memory stores real numbers in 4 words, with 7 digits of accuracy over a range from +/-1.1754944e-38 to +/-3.4028237e38. Floating point memory is stored in F8:xx by default, but other floating point numbers can be stored in other locations. Bit level access is not permitted (or useful) for these numbers.

14.5 SUMMARY

- Program files store users programs in files 2 - 999.
- Data files are available to users and will be 0-999 locations long.
- Default data types on a PLC-5 include Output (O0:), Input (I1:), Status (S2:), Bit (B3:), Timer (T4:), Counter (C5:), Control (R6:), Integer (N7:) and Float (F8:).
- Other memory types include Block Transfer (BT), ASCII (A), ASCII String (ST), BCD (D), Message (MG), PID Control (PD), SFC Status (SC).

- In memory locations a '/' indicates a bit, '.' indicates a word.
- Indirect addresses will substitute memory values between '[', ']'.
- Files are like arrays and are indicated with '#'.
- Expressions allow equations to be typed in.
- Literal values for binary and hexadecimal values are followed by *B* and *H*.

14.6 PRACTICE PROBLEMS

1. Can PLC outputs can be set with Bytes instead of bits?

2. How many types of memory can a PLC-5 have?

3. What are the default program memory locations?

4. How many types of number bases are used in PLC memory?

5. How are timer and counter memory similar?

6. What types of memory cannot be changed?

7. Develop Ladder Logic for a car door/seat belt safety system. When the car door is open, or the seatbelt is not done up, a buzzer will sound for 5 seconds if the key has been switched on. A cabin light will be switched on when the door is open and stay on for 10 seconds after it is closed, unless a key has started the ignition power.

8. Look at the manuals for the status memory in your PLC and find the first scan location

9. Write ladder logic for the following problem description. When button *A* is pressed a value of 1001 will be stored in *N7:0*. When button *B* is pressed a value of -345 will be stored in *N7:1*, when it is not pressed a value of 99 will be stored in *N7:1*. When button *C* is pressed *N7:0* and *N7:1* will be added, and the result will be stored in *N7:2*.

10. Using the status memory locations, write a program that will flash a light for the first 15 seconds after it has been turned on. The light should flash once a second.

11. How many words are required for timer and counter memory?

14.7 PRACTICE PROBLEM SOLUTIONS

1. yes, for example the output word would be addressed as O:000

2. There are 13 different memory types, 10 of these can be defined by the user for data files

between 3 and 999.

3. Program files 0 and 1 are reserved for system functions. File 2 is the default ladder logic program, and files 3 to 999 can be used for other programs.

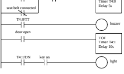
4. binary, octal, BCD, 2s compliment, signed binary, floating point, bits, hexadecimal

5. both are similar. The timer and counter memories both use words for the accumulator and presets, and they use bits to track the status of the functions. These bits are somewhat different, but parallel in function.

6. Inputs cannot be changed by the program, and some of the status bits/words cannot be changed by the user.

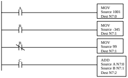
7.

Inputs	Outputs
door open seat belt connected key on	buzzer light

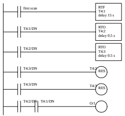


8. S2:1/14 for micrologix, S2:1/15 for PLC-5.

9.


A
MOV
Source 1001
Dest N7:0
B
MOV
Source -345
Dest N7:1
B
MOV
Source 99
Dest N7:1
C
ADD
Source A N7:0
Source B N7:1
Dest N7:2

10.



11. three memory words are used for a timer or a counter.

14.8 ASSIGNMENT PROBLEMS

1. Briefly list and describe the different methods for addressing values (e.g., word, bit, literal, etc.).
2. Could timer 'T' and counter 'C' memory types be replaced with control 'R' memory types? Explain your answer.