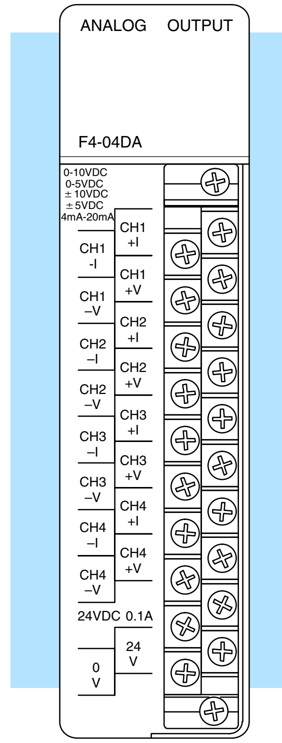
F4–04DA 4-Channel Analog Output

In This Chapter. . . .

Module Specifications

The F4–04DA 4-channel Analog Output module provides several features and benefits.

- It provides four channels of single-ended voltage or current outputs.
- Analog outputs are optically isolated from PLC logic components.
- The module has a removable terminal block, so the module can be easily removed or changed without disconnecting the wiring.
- All four analog outputs may be set in one CPU scan (DL440 and DL450 CPUs only).



Analog Output Configuration Requirements

The F4–04DA Analog Output appears as a 16-point discrete output module. The module can be installed in any slot of a DL405 system, including remote bases. The limitations on the number of analog modules are:

- For local and expansion systems, the available power budget and discrete I/O points.
- For remote I/O systems, the available power budget and number of remote I/O points.

Check the user manual for your particular model of CPU for more information regarding power budget and number of local or remote I/O points.

The following table provides the specifications for the F4–04DA Analog Output Module. Review these specifications to ensure the module meets your application requirements.

Output Specifications

Number of Channels	4
Output Ranges	0–5V, 0–10V, ±5V, ±10V, 4–20 mA
Resolution	12 bit (1 in 4096)
Conversion Method	Successive Approximation
Output Type	Single ended, 1 common
Output Impedance	0.2Ω typical, voltage output
Load Impedance	2KΩ minimum, voltage output 0Ω minimum, current output
Maximum Load / Voltage	680Ω/18V, 1KΩ/24V, 1.5KΩ/36V, current output
Voltage Output Current	5 mA sink or source
Short-Circuit Current	15 mA typical, voltage output
Linearity Error	±1 count (±0.025%) maximum
Gain Calibration Error	±8 counts maximum, voltage output –8 to +11 counts maximum, current output
Offset Calibration Error	±2 counts maximum, voltage output –5 to +9 counts maximum, current output
Conversion Time	5 µs maximum, settling time 0.3 ms maximum, digital out to analog out

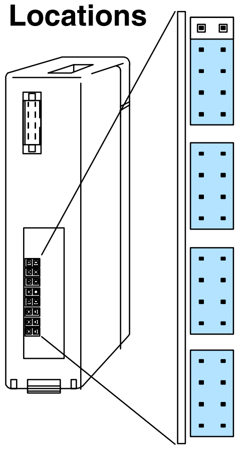
General Module Specifications

Digital Output Points Required	16 point (Y) outputs, 12 bits binary data, 4 channel select bits
Power Budget Requirement	120 mA @ 5 VDC (from base)
External Power Supply	24 VDC, 100 mA, class 2 ±10% (add 20 mA for each current loop used)
Accuracy vs. Temperature	±50 ppm / °C maximum full scale ±25 ppm / °C maximum offset
Operating Temperature	0 to 60°C (32 to 140°F)
Storage Temperature	–20 to 70°C (–4 to 158°F)
Relative Humidity	5 to 95% (non-condensing)
Environmental air	No corrosive gases permitted
Vibration	MIL STD 810C 514.2
Shock	MIL STD 810C 516.2
Noise Immunity	NEMA ICS3-304

One count in the specification table is equal to one least significant bit of the analog data value (1 in 4096).

Setting the Module Jumpers

Jumper Locations

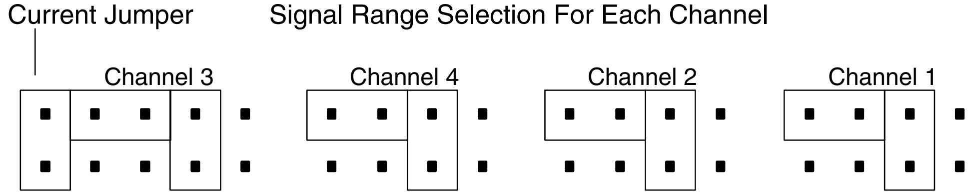


If you examine the rear of the module, you will notice several jumpers. These jumpers are used to select the signal range for each channel. There is also a current jumper. Some of the ranges can be selected with the current jumper installed or removed. To prevent losing jumpers when they are removed, a good place to store them is to reinstall each removed jumper over a single pin.

The signal range choices include five commonly encountered signal ranges: ±5 VDC, ±10 VDC, 4 – 20 mA, 0 – 5 VDC, and 0 – 10 VDC. The jumper settings for these signal ranges are shown in the table below.

In addition, the module supports some additional ranges that are not encountered very often. Use the last table in this section for those jumper settings.

The module is set at the factory for a 4–20 mA signal on all four channels. If this is acceptable you do not have to change any of the jumpers. The following diagram shows how the jumpers are set from the factory.



Selecting Common Output Signal Ranges

If you are using any 4–20 mA signals, you must have the current jumper installed. You can still select some of the other voltage ranges (except for the ±10 VDC range, which requires that the current jumper be removed). The following table shows the jumper selections for the commonly used ranges.

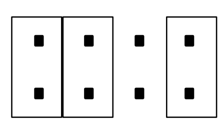
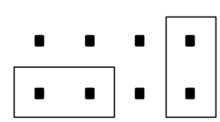
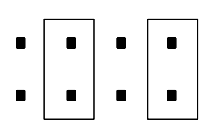
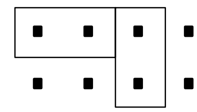
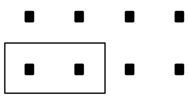
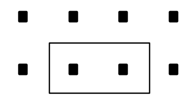
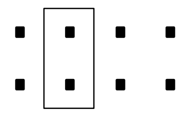
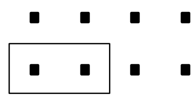
Bipolar Signal Range	Current Jumper Installed	Current Jumper Removed
±5 VDC		
±10 VDC	Not supported with current jumper installed	

Table is continued on the next page.

Commonly Used Signal Ranges (continued)

Unipolar Signal Range	Current Jumper Installed	Current Jumper Removed
4 to 20 mA		Not supported with current jumper removed
0 to +5 VDC		
0 to +10 VDC		

Selecting Special Signal Ranges

The F4–04DA module supports additional ranges that are not encountered very often. The following table shows the jumper selections for these ranges.

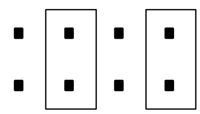
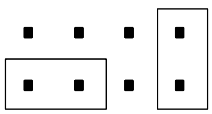
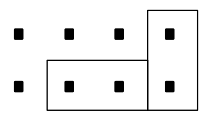
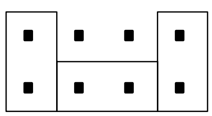
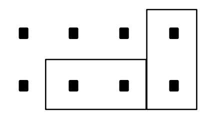
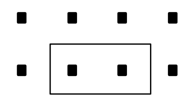
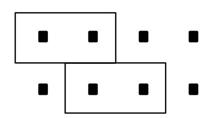
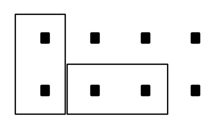
Signal Range	Current Jumper Installed	Current Jumper Removed
–10 to 0 VDC		Not supported with current jumper removed
–5 to 0 VDC		Not supported with current jumper removed
–2.5 to 0 VDC		Not supported with current jumper removed
± 1.25 VDC		Not supported with current jumper removed

Table is continued on the next page.

Special Signal Ranges (continued)

Signal Range	Current Jumper Installed	Current Jumper Removed
±2.5 VDC	Not supported with current jumper installed	
0 to +2.5 VDC		Not supported with current jumper removed
+1.25 to +6.25 VDC	Not supported with current jumper installed	
+2.5 to +7.5 VDC	Not supported with current jumper installed	

Connecting the Field Wiring

Wiring Guidelines

Your company may have guidelines for wiring and cable installation. If so, you should check those before you begin the installation. Here are some general things to consider.

- Use the shortest wiring route whenever possible.
- Use shielded wiring and ground the shield at the module or the power supply return (0V). *Do not* ground the shield at both the module and the transducer.
- Do not run the signal wiring next to large motors, high current switches, or transformers. This may cause noise problems.
- Route the wiring through an approved cable housing to minimize the risk of accidental damage. Check local and national codes to choose the correct method for your application.

User Power Supply Requirements

The F4–04DA requires a separate power supply. The DL430/440/450 CPUs, D4–RS Remote I/O Controller, and D4–EX Expansion Units have built-in 24 VDC power supplies that provide up to 400mA of current. Depending on the number of modules and types of output signals used, you can use this power source instead of a separate supply. If you would rather use a separate supply, choose one that meets the following requirements: 21.6–26.4 VDC, Class 2, 500mA current. If you are using current loops, add 20 mA per current loop.

Load Requirements

Each channel can be wired independently for voltage or current. However, you cannot use both ±10 VDC and 4–20 mA signals on the same module.

- Current loads must have an impedance between 0 and 1000 ohms.
- Voltage loads must have an impedance greater than 2K ohms.

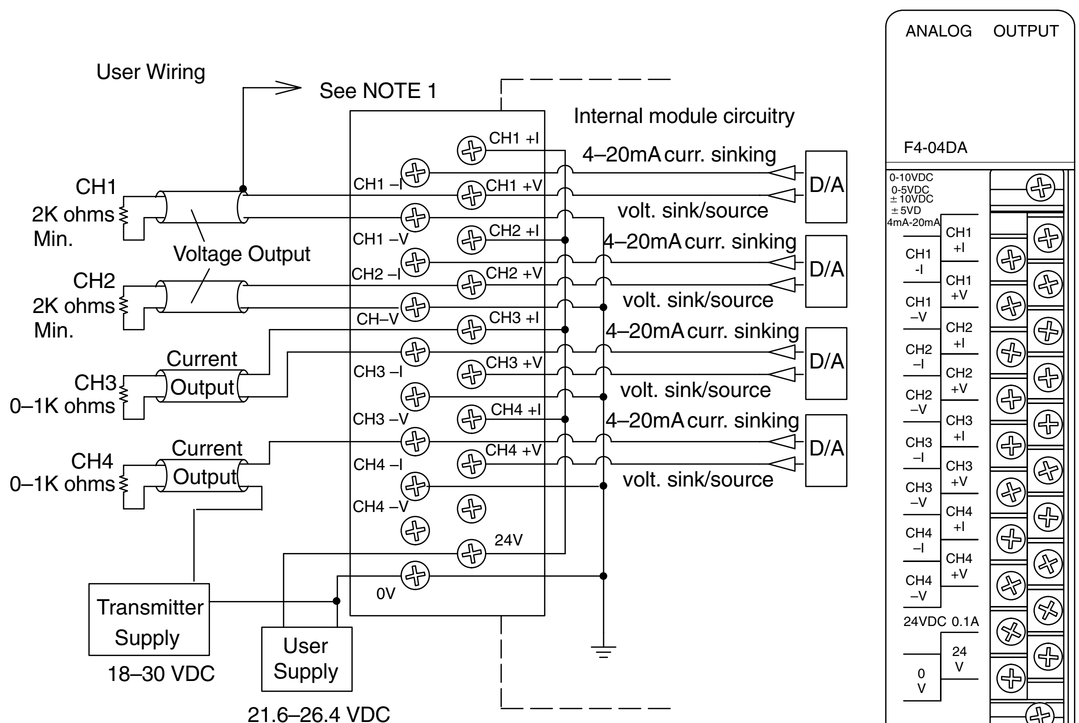
Removable Connector

The F4–04DA module has a removable connector to make wiring easier. Simply remove the retaining screws and gently pull the connector from the module.

Wiring Diagram

NOTE 1: Shields should be connected to the 0V terminal of the module or power supply.

NOTE 2: Unused voltage and current outputs should remain open (no connections).

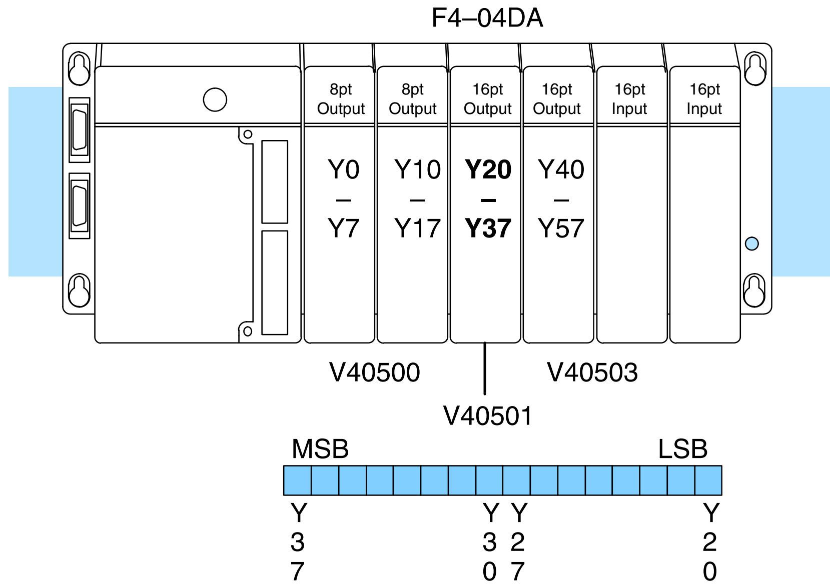


Module Operation

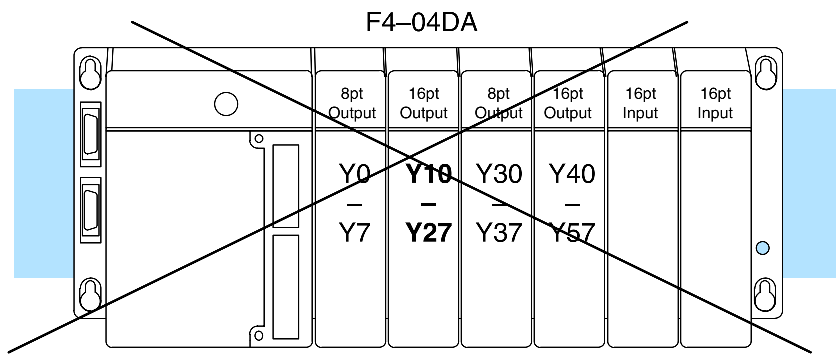
DL430 Special Requirements

Even though the module can be placed in any slot, it is important to examine the configuration if you are using a DL430 CPU. As you will see in the section on writing the program, you use V-memory locations to send the analog data. As shown in the following diagram, if you place the module so the output points do not start on a V-memory boundary, the instructions cannot access the data.

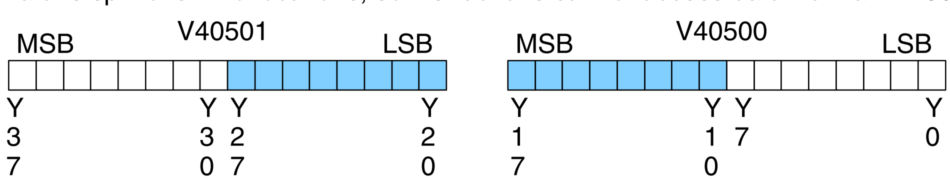
Correct!



Wrong!



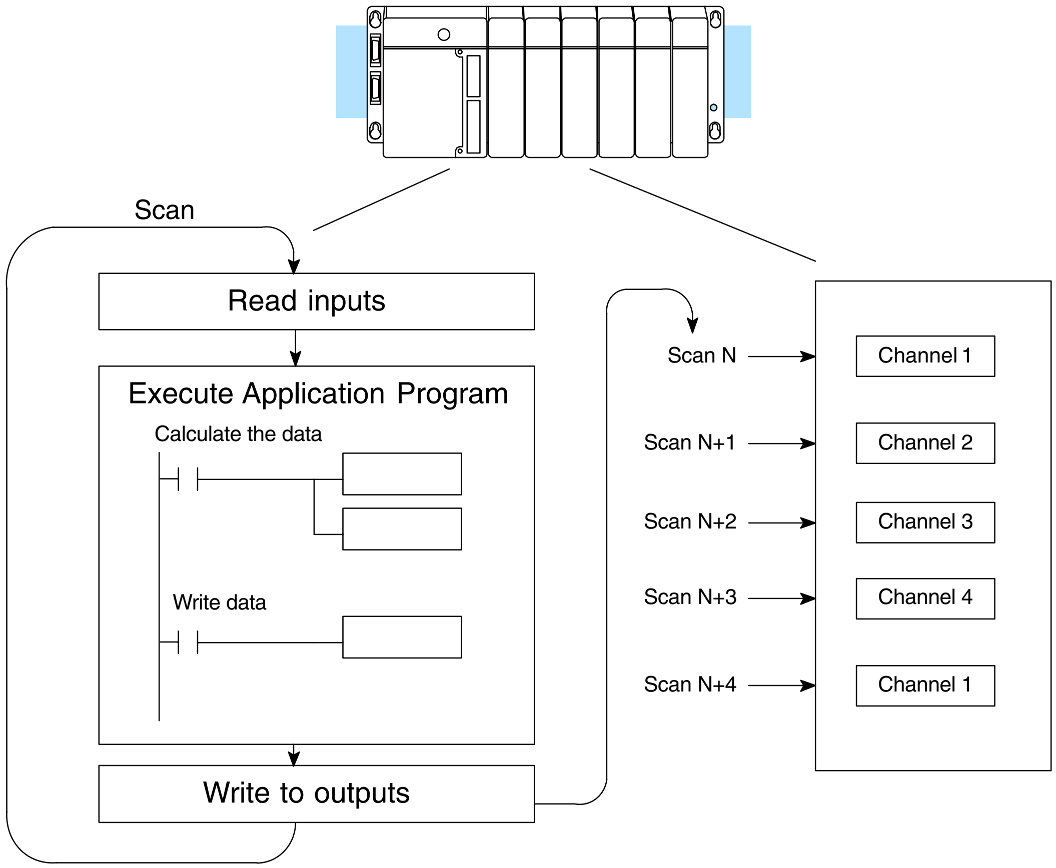
Data is split over two locations, so instructions cannot access data from a DL430.



Before you begin writing the control program, it is important to take a few minutes to understand how the module processes and represents the analog signals.

Channel Scanning Sequence

The F4–04DA module allows you to update the channels in any order. Your control program determines which channel gets updated on any given scan by *turning off* a bit that is associated with each channel. With a DL440 or DL450 CPU, you can use immediate instructions to update all four channels in the same scan (we will show you how to do this later).

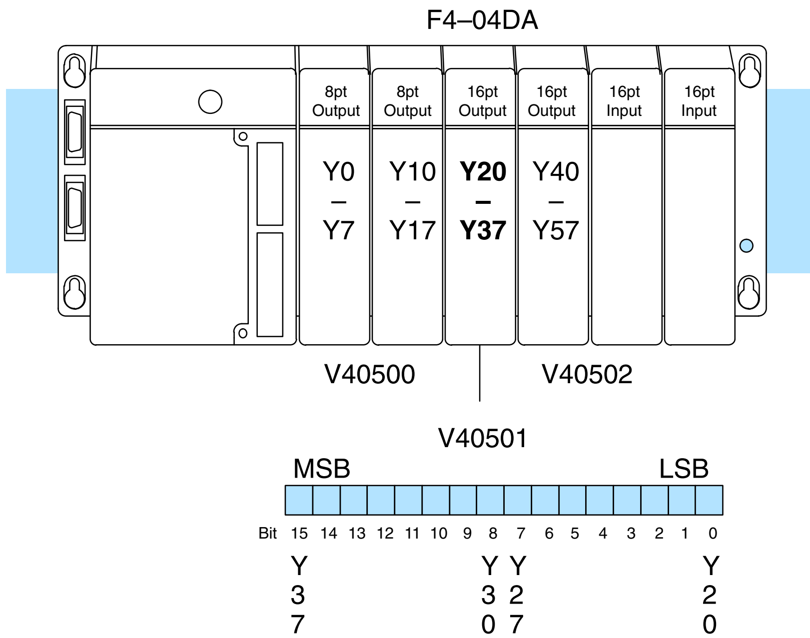


Output Bit Assignments

You may recall the F4–04DA module requires 16 discrete output points from the CPU. These points provide:

- The digital representation of the analog signal.
- Identification of the channel that is to receive the data.

Since all output points are automatically mapped into V-memory, it is very easy to determine the location of the data word that will be assigned to the module.



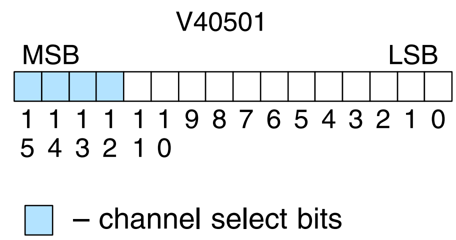
Within this V-memory location the individual bits represent specific information about the analog signal.

Channel Select Bits

The last four bits (outputs) select the channel that will be updated with the data. The bits are assigned as follows.

Bit	Y point	Channel
12	Y34	1
13	Y35	2
14	Y36	3
15	Y37	4

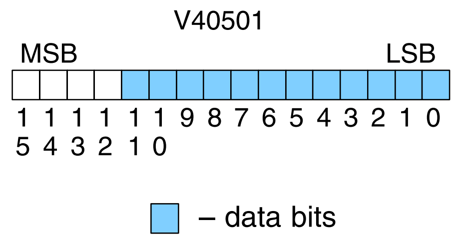
When the output is off, the data currently in the accumulator is sent to the corresponding channel. If the bit is on, the corresponding channel holds the last value that was received from the CPU.



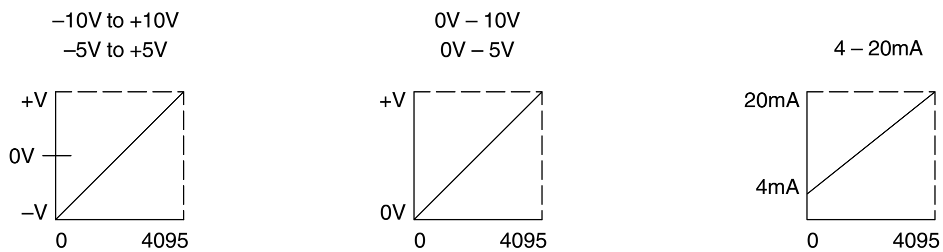
Analog Data Bits

The first twelve bits of the V-memory location represent the analog data in binary format.

Bit	Value	Bit	Value
0	1	6	64
1	2	7	128
2	4	8	256
3	8	9	512
4	16	10	1024
5	32	11	2048



Since the module has 12-bit resolution, the analog signal is made of 4096 “pieces” ranging from 0 – 4095 (2^{12}). For example, with a 0 to 10V scale, you would send a 0 to get a 0V signal, and 4095 to get a 10V signal. This is equivalent to a binary value of 0000 0000 0000 to 1111 1111 1111, or 000 to FFF hexadecimal. The following diagram shows how this relates to each signal range.



Each “count” can also be expressed in terms of the signal level by using the equation shown. The following table shows the smallest signal change that occurs when the digital value is increased.

$$\text{Resolution} = \frac{H - L}{4095}$$

H = high limit of the signal range

L = low limit of the signal range

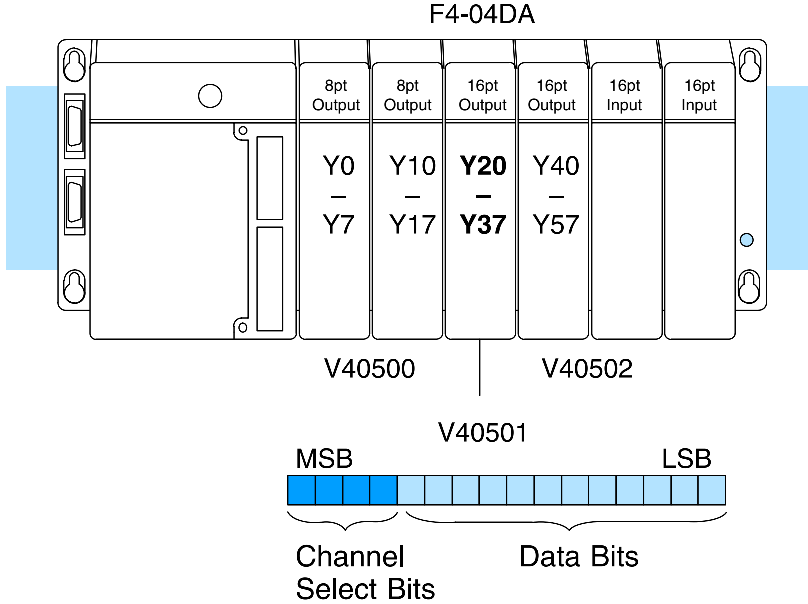
Signal Range	Span (H – L)	Divide By	Smallest Change
± 10V	20V	4095	4.88 mV
± 5V	10V	4095	2.44 mV
0 to 5V	5V	4095	1.22 mV
0 to 10V	10V	4095	2.44 mV
4 to 20mA	16mA	4095	3.91 μA

Now that you understand how the analog signal is represented by the digital value, you're ready to write the control program.

Writing the Control Program

Update Any Channel

As mentioned earlier, you can update any channel or any channels during the same scan. The following diagram shows the data locations for an example system. You use the channel selection outputs to determine which channel gets updated (more on this later).



Calculating the Digital Value

Your program has to calculate the digital value to send to the analog module. There are many ways to do this, but almost all applications are understood more easily if you use measurements in engineering units. This is accomplished by using the conversion formula shown.

You may have to make adjustments to the formula depending on the scale you choose for the engineering units.

$$A = U \frac{4095}{H - L}$$

A = analog value (0 – 4095)

U = engineering units

H = high limit of the engineering unit range

L = low limit of the engineering unit range

Consider the following example which controls pressure from 0.0 to 99.9 PSI. By using the formula, you can easily determine the digital value that should be sent to the module. The example shows the conversion required to yield 49.4 PSI. Notice the formula uses a multiplier of 10. This is because the decimal portion of 49.4 cannot be loaded, so you adjust the formula to compensate for it.

$$A = 10U \frac{4095}{10(H - L)}$$

$$A = 494 \frac{4095}{1000 - 0}$$

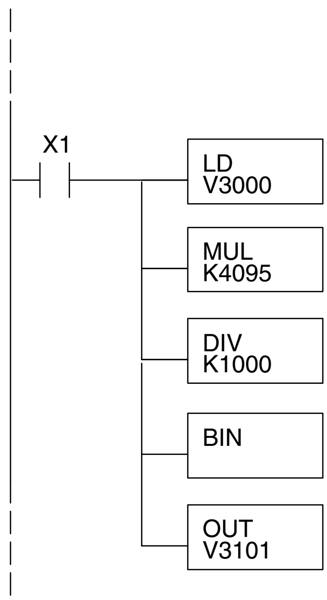
$$A = 2023$$

Engineering Unit Conversion

✓ 430 ✓ 440 ✓ 450

Here is how you would write the program to perform the engineering unit conversion. This example assumes you have calculated or loaded the engineering unit value and stored it in V3000. Also, you have to perform this for all four channels if you are using different data for each channel.

NOTE: The DL405 offers various instructions that allow you to perform math operations using binary, BCD, etc. It is usually easier to perform any math calculations in BCD and then convert the value to binary before you send the data to the module. If you are using binary math, you do not have to include the BIN conversion.



When X1 is on, the engineering units (stored in V3000) are loaded into the accumulator. This example assumes the numbers are BCD.

Multiply the accumulator by 4095 (to start the conversion).

Divide the accumulator by 1000 (because we used a multiplier of 10, we have to use 1000 instead of 100).

Convert the BCD number to binary.

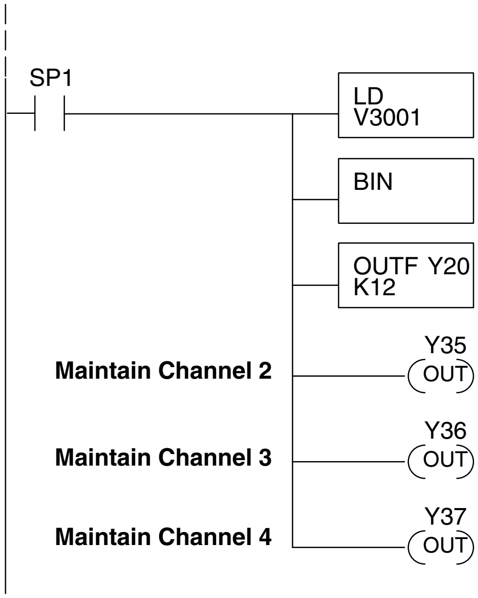
Store the result in V3101. This is the digital value, in binary form, that should be sent to the module.

Sending Data to One Channel

The following programs show you how to update a single channel. Notice the DL440 and DL450 CPUs require slightly different programs than the DL430 CPU. Since the DL430 does not support the OUTF instruction, the program must be modified to make sure the channel select bits are not accidentally changed by the data in the accumulator. The DL430 example will also work with DL440 and DL450 CPUs. This example assumes you already have the data loaded in V3001.

430 ✗ 440 ✓ 450 ✓

DL440/450 Example



The LD instruction loads the data for channel 1 into the accumulator. Since SP1 is used, this rung automatically executes on every scan. You could also use an X, C, etc. permissive contact.

The BIN instruction converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere).

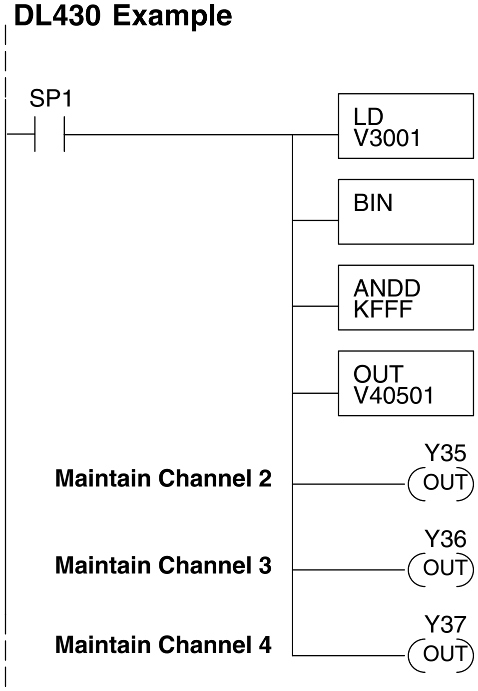
The OUTF sends the 12 bits to the data word. Our example starts with Y20, but the actual value depends on the location of the module in your application.

When the output selection bit is on, the module maintains the channel at its current value. If the output selection bit is off, the corresponding channel is updated with the data from the accumulator. This example assumes that Y34 is off and not used elsewhere in the program. By turning Y35 – Y37 on, only channel 1 gets updated. See the table below.

Output	Channel
Y34	Ch. 1
Y35	Ch. 2
Y36	Ch. 3
Y37	Ch. 4

430 ✓ 440 ✓ 450 ✓

DL430 Example



The LD instruction loads the data for channel 1 into the accumulator. Since SP1 is used, this rung automatically executes every scan. You could also use an X, C, etc. permissive contact.

The BIN instruction converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere).

The ANDD instruction masks off the channel select bits to prevent an accidental channel selection.

The OUT instruction sends the data to the module. Our example starts with V40501, but the actual value depends on the location of the module in your application.

When the output selection bit is on, the module maintains the channel at its current value. If the output selection bit is off, the corresponding channel is updated with the data from the accumulator. This example assumes that Y34 is off and not used elsewhere in the program. By turning Y35 – Y37 on, only channel 1 gets updated. See the table below.

Output	Channel
Y34	Ch. 1
Y35	Ch. 2
Y36	Ch. 3
Y37	Ch. 4

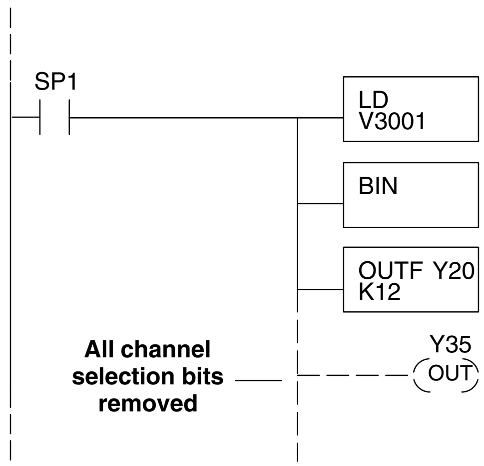
Sending the Same Data to All Channels

✗ 430 ✓ 440 ✓ 450

The following programs show you how to update all channels with the same data. The primary difference from the previous example programs is that you do not have to include the channel selection bits. By leaving the selection bits off, the module uses the same data to update all the channels.

Notice the DL440 and DL450 CPUs require slightly different programs than the DL430 CPU. Since the DL430 does not support the OUTF instruction, the program must be modified to make sure the channel select bits are not accidentally changed by the data in the accumulator. The DL430 example will also work with DL440 and DL450 CPUs. This example assumes you already have the data loaded in V3001.

DL440/450 Example



The LD instruction loads the data for channel 1 into the accumulator. Since SP1 is used, this rung automatically executes on every scan. You could also use an X, C, etc. permissive contact.

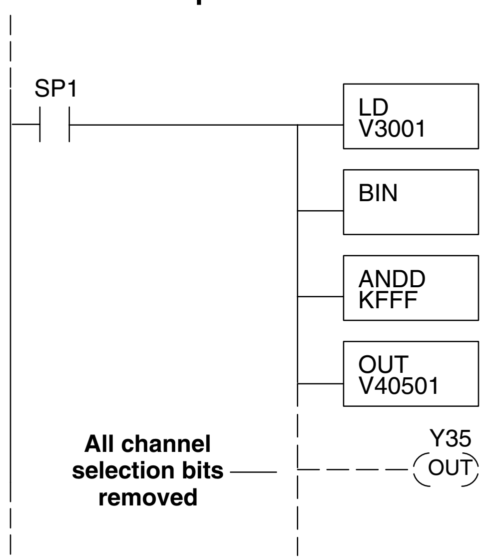
The BIN instruction converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere).

The OUTF sends the 12 bits to the data word. Our example starts with Y20, but the actual value depends on the location of the module in your application.

Y34 – Y37 must be off.

✓ 430 ✓ 440 ✓ 450

DL430 Example



The LD instruction loads the data for channel 1 into the accumulator. Since SP1 is used, this rung automatically executes on every scan. You could also use an X, C, etc. permissive contact.

The BIN instruction converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere).

The ANDD instruction masks off the channel select bits to prevent an accidental channel selection.

The OUT instruction sends the data to the module. Our example starts with V40501, but the actual value depends on the location of the module in your application.

Y34 – Y37 must be off.

Sequencing the Channel Updates

The next four example programs show you how to send digital values to the module when you have more than one channel. These examples will automatically update all four channels over four scans.

The first two sequencing examples, examples 1 and 2, are fairly simple and will work in most situations. We recommend these for new users. They use control relays C1 through C4 as index numbers corresponding to the channel updated on any particular scan. At the end of each scan, only one control relay C1 through C4 is on. On each subsequent scan, the next control relay energizes. The channel sequencing automatically begins with channel 1 on the first scan, or after any disruption in the logic. You must use example 1 with DL430 CPUs. Either example will work with DL440 or DL450 CPUs.

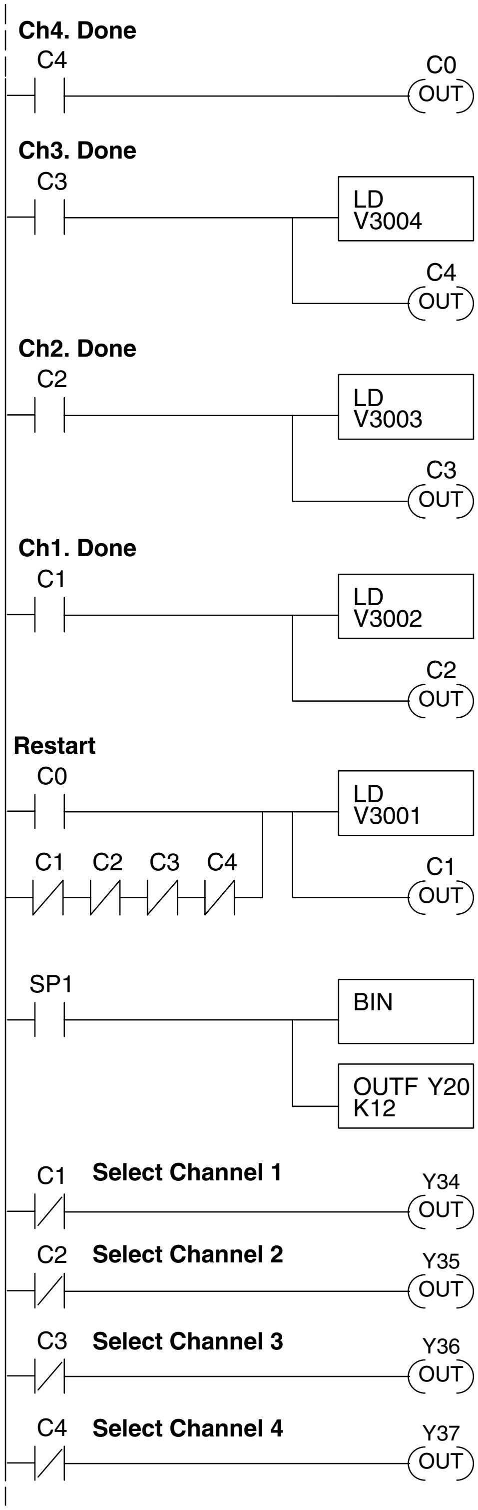
The next two examples, 3 and 4, are slightly more complex. However, they do not depend on the use of control relays to provide channel sequencing. Instead, they use function boxes to increment a channel pointer value in V-memory. Then, other instructions perform bit manipulations to properly position the channel select bits in the output word to the module. You must use example 3 with DL430 CPUs. Either example will work with DL440 or DL450 CPUs.

In the last example, we show you how to update all four channels in the same scan with DL440 and DL450 CPUs. However, this increases the scan time and you may not always need to update all four channels on every scan.

Sequencing Example 1, DL440/450

430 440 450

The following program example shows how to send digital values to the module when you have more than one channel. This example assumes you already have the data loaded in V3001, V3002, V3003, and V3004 for channels 1 – 4 respectively. It is important to use the rungs in the order shown for the program to work. This example will not work with DL430 CPUs.



When channel 4 has been updated, C0 restarts the update sequence.

When channel 3 has been updated, this rung loads the data for channel 4 into the accumulator. By turning on C4, this triggers the channel update (see the channel select rungs below).

When channel 2 has been updated, this rung loads the data for channel 3 into the accumulator. By turning on C3, this triggers the channel update (see the channel select rungs below).

When channel 1 has been updated, this rung loads the data for channel 2 into the accumulator. By turning on C2, this triggers the channel update (see the channel select rungs below).

This rung loads the data for channel 1 into the accumulator. C0 restarts the sequence after channel 4 is done (see the top rung). The first scan or any interruption in control relay sequencing is detected when control relays C1 through C4 are off. In this case, we also start the sequence with channel 1.

This rung converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere). It also loads the data to the appropriate bits of the data word. Our example starts with Y20, but the actual value depends on the location of the module in your application.

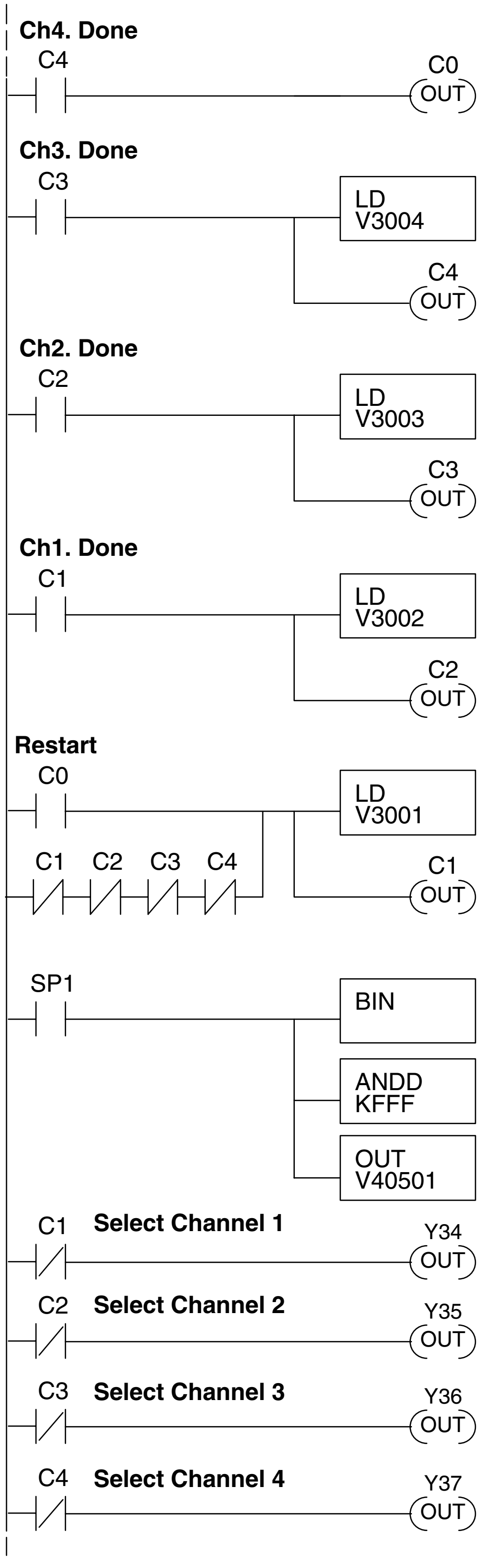
When the output selection bit is on (Y34–Y37 in this case), the module maintains the channel at its current value. If the output selection bit is off, the corresponding channel is updated with the data from the accumulator.

Output	Channel
Y34	Ch. 1
Y35	Ch. 2
Y36	Ch. 3
Y37	Ch. 4

Sequencing Example 2, DL430

✓ 430 ✓ 440 ✓ 450

Since the DL430 does not support the OUTF instruction, the previous program must be modified to make sure the channel select bits are not accidentally changed by the data in the accumulator. It is important to use the rungs in the order shown for the program to work. This example will also work with DL440 and DL450 CPUs.



When channel 4 has been updated, C0 restarts the update sequence.

When channel 3 has been updated, this rung loads the data for channel 4 into the accumulator. By turning on C4, this triggers the channel update (see the channel select rungs below).

When channel 2 has been updated, this rung loads the data for channel 3 into the accumulator. By turning on C3, this triggers the channel update (see the channel select rungs below).

When channel 1 has been updated, this rung loads the data for channel 2 into the accumulator. By turning on C2, this triggers the channel update (see the channel select rungs below).

This rung loads the data for channel 1 into the accumulator. C0 restarts the sequence after channel 4 is done (see the top rung). The first scan or any interruption in control relay sequencing is detected when control relays C1 through C4 are off. In this case, we also start the sequence with channel 1.

This rung converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere). The ANDD instruction masks off the channel select bits to prevent an accidental channel selection. The OUT instruction sends the data to the module. Our example starts with V40501, but the actual value depends on the location of the module in your application.

When the output selection bit is on (Y34 – Y37 in this case), the module maintains that channel at its current value. If the output selection bit is off, the corresponding channel is updated with the data from the accumulator.

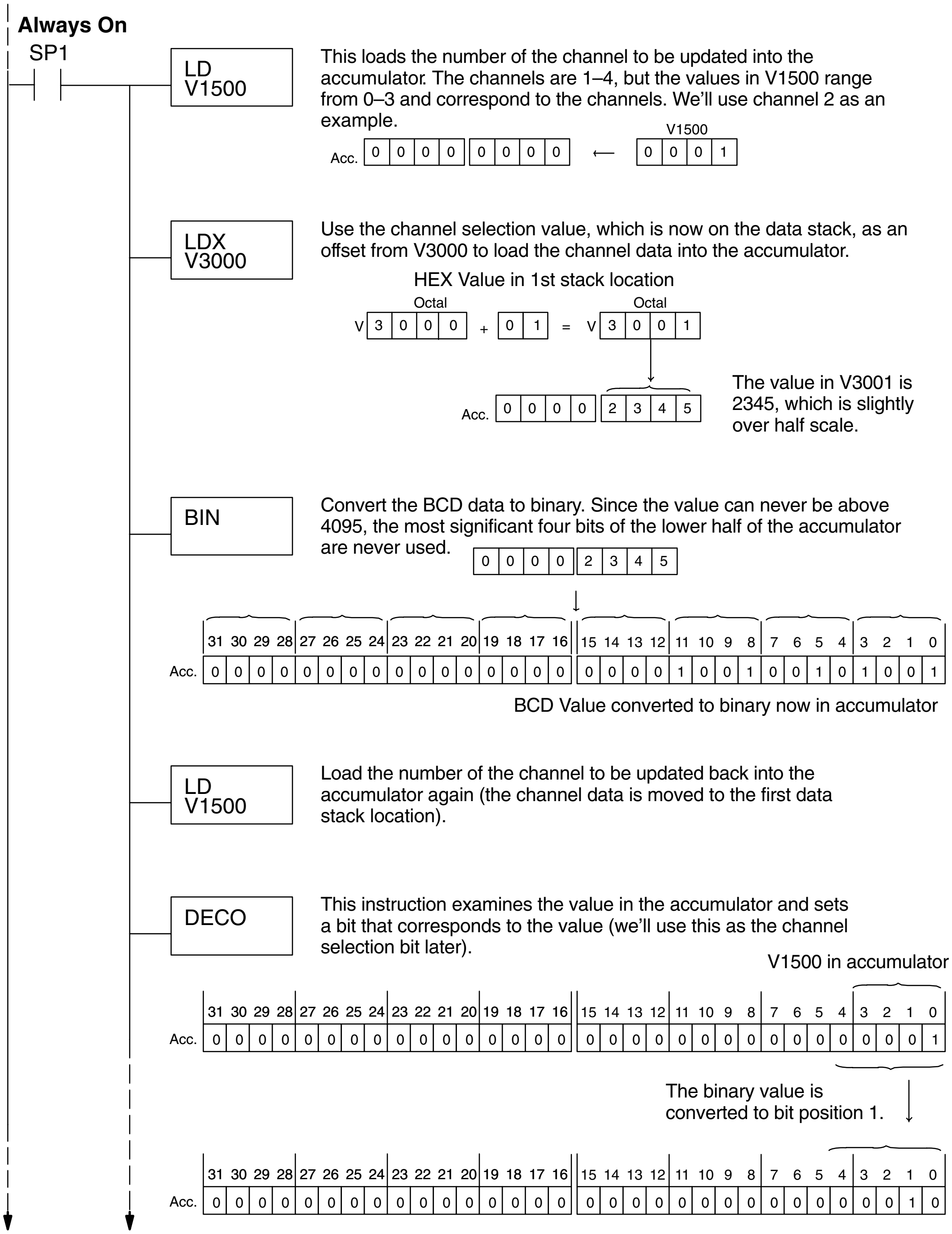
Output	Channel
Y34	Ch. 1
Y35	Ch. 2
Y36	Ch. 3
Y37	Ch. 4

Sequencing Example 3, DL440/450

430 440 450

The following program example shows how to send digital values to the module when you have more than one channel. This example will not work with DL430 CPUs. This example assumes you are using the following data locations.

- V3000 – channel 1 data V3001 – channel 2 data
- V3002 – channel 3 data V3004 – channel 4 data
- V1500 – channel to update: 0 = ch. 1, 1 = ch. 2, 2 = ch. 3, 3 = ch. 4



rogram is continued on the next page.

Example 3 Continued

INV

This inverts the accumlator bits. Remember, this module activates the channels when they are off, so the channel selection bit needs to be a zero. That's why we use the invert instruction.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1

SHFL K12

This instruction moves the channel selection bit into the proper location. We'll use it later when we send the 16-bit data word to the module.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

ADDBS

Earlier in the program the data value was placed into the first data stack location. The ADDBS instruction adds the value currently in the lower 16 bits of the accumulator to the value in the first data stack location.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

+

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Stack	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	1

↓

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	0	0	1	0	1	0	0	1

Data for Analog Module

OUTF Y20 K16

Sends the first 16 bits stored in the accumulator to the analog module. The most significant four bits of the analog word contain the channel selection bits. The remaining 12 bits contain the analog data.

INCB V1500

Increments the channel selection value. This allows the logic to cycle through all four channels.

V1500 K3 ─┤ = ├─ **LD K0**

When channel 4 has been updated, then reset the channel selection memory location to 0 (remember, 0 is for channel 1).

OUT V1500

Sequencing Example 4, DL430

✓ 430 ✓ 440 ✓ 450

The following program example shows how to send digital values to the module when you have more than one channel. This example will also work with DL440 and DL450 CPUs. This example assumes you are using the following data locations.

- V3000 – channel 1 data   V3001 – channel 2 data
- V3002 – channel 3 data   V3004 – channel 4 data
- V1500 – channel to update: 0 = ch. 1, 1 = ch. 2, 2 = ch. 3, 3 = ch. 4
- V1501 – temporary location for the channel selection

Always On
SP1

LD V1500

This loads the number of the channel to be updated into the accumulator. The channels are 1–4, but the values in V1500 range from 0–3 and correspond to the channels. We'll use channel 2 as an example.

Acc. 0 0 0 0 | 0 0 0 0 ⟵ V1500 0 0 0 1

DECO

This instruction examines the value in the accumulator and sets a bit that corresponds to the value (we'll use this as the channel selection bit later).

V1500 in accumulator

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

The binary value is converted to bit position 1.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

INV

This inverts the accumulator bits. Remember, this module activates the channels when they are off, so the channel selection bit needs to be a zero. That's why we use the invert instruction.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1

SHFL K12

This instruction moves the channel selection bit into the proper location. We'll use it later when we send the 16-bit data word to the module.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

OUT V1501

Store the channel selection portion of the data word in V1501 temporarily (we'll use it again later).

LD V1500

Again load the channel selection from V1500.

Acc. 0 0 0 0 | 0 0 0 0 ⟵ V1500 0 0 0 1

Program is continued on the next page.

Example 4 Continued

LDX V3000

Use the channel selection value, which is now on the data stack, as an offset from V3000 to load the channel data into the accumulator.

HEX Value in 1st stack location

V 3000 (Octal) + 01 = V 3001 (Octal)

Acc. 0 0 0 0 2 3 4 5

The value in V3001 is 2345, which is slightly over half scale.

BIN

Convert the BCD data to binary. Since the value can never be above 4095, the most significant four bits of the lower half of the accumulator are never used.

0 0 0 0 2 3 4 5

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	1

BCD Value converted to binary now in accumulator

OR V1501

Earlier in the program the channel selection portion of the data word was created and stored in V1501. Now we can OR this location with the data word currently in the accumulator to get the final data word that is ready to be sent to the analog module.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	1

\+

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
V1501	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Acc.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	0	0	1	0	1	0	0	1

Data for Analog Module

OUT V40501

Sends the data stored in the lower half of the accumulator to the analog module (the OUT instruction ignores the upper 16 bits of the accumulator). The most significant four bits of the analog word contain the channel selection bits. The remaining 12 bits contain the analog data.

INCB V1500

Increments the channel selection value. This allows the logic to cycle through all four channels.

V1500 = K4

LD K0

When channel 4 has been updated, this instruction resets the channel selection memory location to 0 (remember, 0 is for channel 1).

OUT V1500

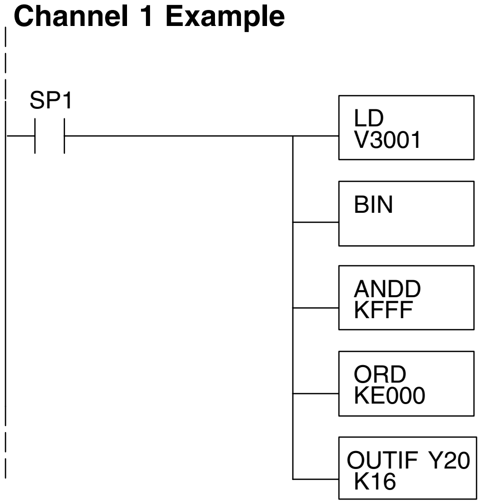
Updating All Channels in a Single Scan, DL440/450

430 440 450

By using the Immediate instructions found in the DL440 and DL450 CPUs, you can easily update all four channels in a single scan. Before choosing this method, remember that it slows the CPU scan time. To minimize this impact, change the SP1 (Always On) contact to an X, C, etc. permissive contact that only updates the channels as required. This example assumes you already have the data loaded in V3001, V3002, V3003, and V3004 for channels 1 – 4 respectively. This example will not work with DL430 CPUs.

NOTE: This program will not work in a remote/slave arrangement. Use one of the programs shown that reads one channel per scan.

Channel 1 Example



The LD instruction loads the data for channel 1 into the accumulator.

The BIN instruction converts the accumulator data to binary (you must omit this step if you've already converted the data elsewhere).

The ANDD instruction masks off the channel select bits to prevent an accidental channel selection.

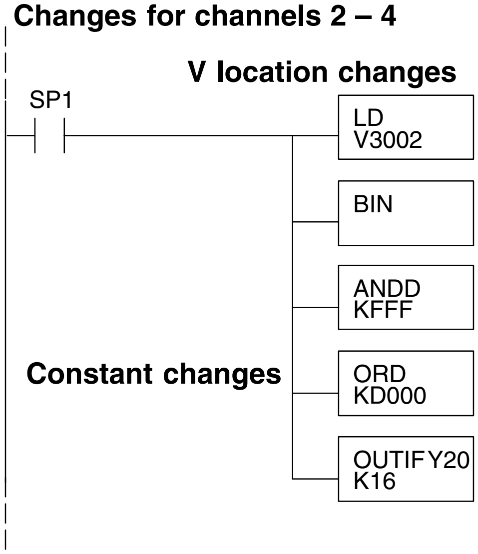
The ORD instruction (with K E000) has the effect of turning on Y35–Y37 and turning off Y34. By leaving Y34 off, channel 1 is updated with the data.

The OUTIF sends 16 bits to the data word. Our example starts with Y20, but the actual value depends on the location of the module in your application.

You have to send 16 bits with the OUTIF instruction. If you don't send all 16 bits, the module will ignore the data.

The remaining channels are updated with a similar program segment. The only changes are the location of the data for each channel (V3002, V3003, and V3004) and the ORD instruction. The constant loaded with the ORD instruction is different for each channel. The following example shows where these differences occur.

Changes for channels 2 – 4



The LD instruction loads the data for channel 1 into the accumulator.

Location	Channel
V3001	1
V3002	2
V3003	3
V3004	4

The ORD instruction has the effect of selecting the appropriate channel to be updated. The following constants are used.

Constant	Channel
K E000	1
K D000	2
K B000	3
K 7000	4

Analog and Digital Value Conversions

Sometimes it is helpful to be able to quickly convert between the voltage or current signal levels and the digital values. This is especially helpful during machine startup or troubleshooting. The following table provides formulas to make this conversion easier.

Range	If you know the digital value ...	If you know the analog signal level ...
0 to 5V	$A = \frac{5D}{4095}$	$D = \frac{4095}{5}(A)$
0 to 10V	$A = \frac{10D}{4095}$	$D = \frac{4095}{10}(A)$
±5V	$A = \frac{10D}{4095} - 5$	$D = \frac{4095}{10}(A + 5)$
±10V	$A = \frac{20D}{4095} - 10$	$D = \frac{4095}{20}(A + 10)$
4 to 20mA	$A = \frac{16D}{4095} + 4$	$D = \frac{4095}{16}(A - 4)$

For example, if you are using the –10 to +10V range and you know you need a 6V signal level, you would use the following formula to determine the digital value that should be stored in the V-memory location that contains the data.

$$D = \frac{4095}{20}(A + 10)$$

$$D = \frac{4095}{20}(6V + 10)$$

$$D = (204.75)\ (16)$$

$$D = 3276$$

Now you have all the necessary information to get your analog module installed and operating correctly.