Errata Sheet

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

Product Family:	DL305	**Date**:	September 2018
Manual Number	D3-ANLG-M		
Revision and Date	3rd Edition, February 2003		

Changes to Chapter 2. D3-04AD 4-Channel Analog Input

This module is no longer available. Please consider the F3-08AD-1 or F3-04ADS as a replacement

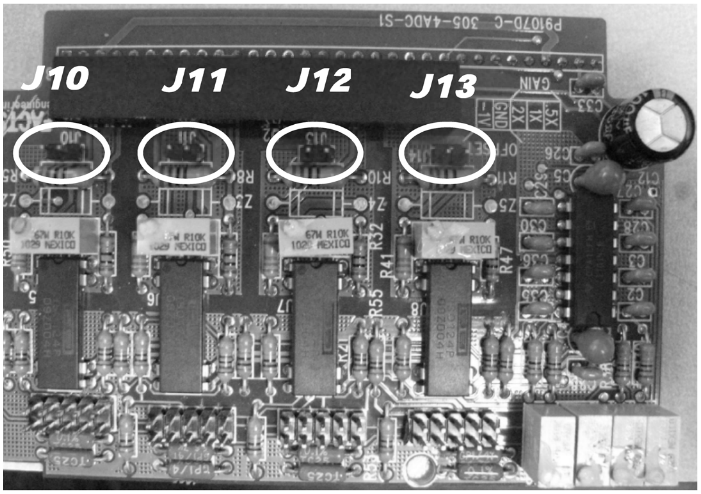
Changes to Chapter 3. F3-04ADS 4-Channel Isolated Analog Input

Page 3-3. Setting the Module Jumpers; Jumper Locations

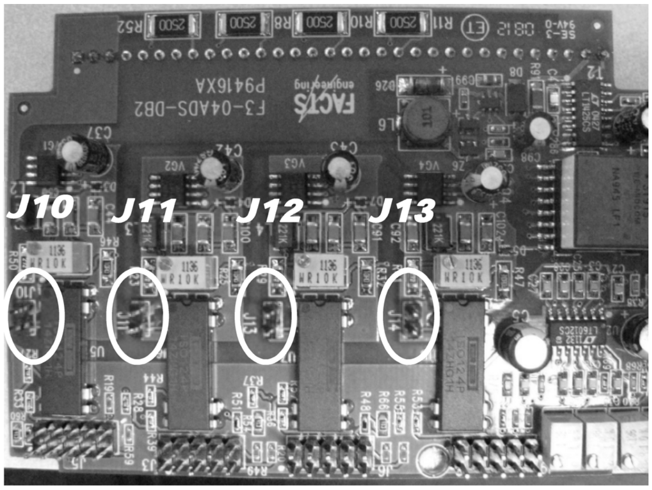
The PC board was redesigned and the locations of jumpers J10, J11, J12, and J13 changed. The jumpers were rotated 90 degrees and are closer to the back of the module than the original layout. The functionality of the jumpers did not change. The orientaton of the 5 pairs of pins for each channel is the same.

The photo on the right shows the new design, while the one on the left shows the original PC board. The photo on the left matches the drawing shown on page 3-3. The redesigned PC boards are in modules manufactured starting in mid-2012.O

Original PC Board Layout
(Manufactured prior to mid-2012)



Redesigned PC Board Layout
(Manufactured after mid-2012)



Errata Sheet

Changes to Chapter 5. F3–16AD 16-Channel Analog Input

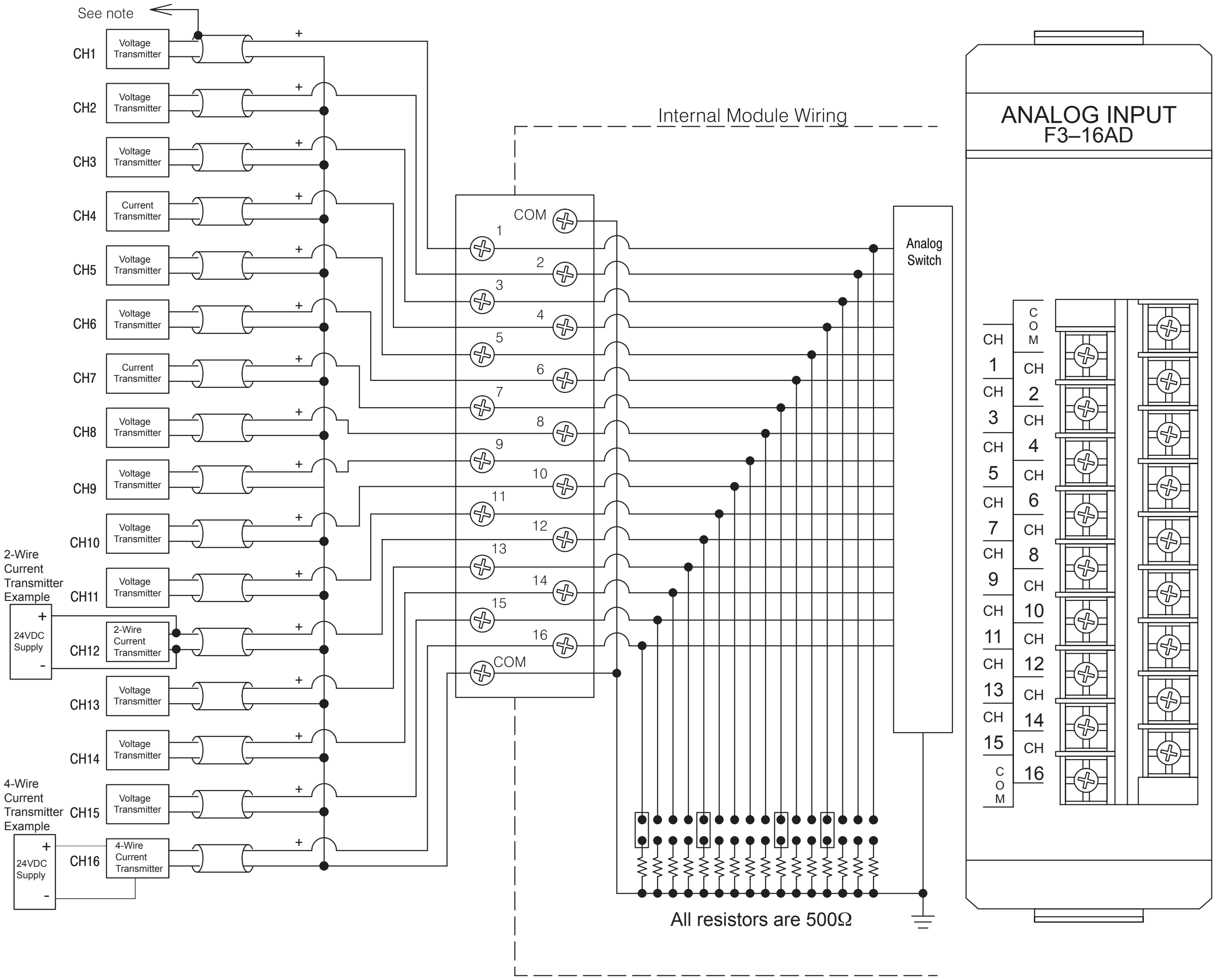
Page 5-9. Wiring Diagram

The wiring diagram shows “current transmitters” CH 4, 7, 12, and 16. The diagram should show external 24VDC power supplies for these current transmitters. A 2-wire current transmitter example of this has been added to the diagram below for CH12.

Also, CH16 has been changed to show a 4-wire current transmitter example.

Wiring Diagram

Note 1: Terminate all shields at their respective signal source.
Note 2: Jumpers for CH4, 7, 12 and 16 are installed for current input.



F3–16AD
16-Channel
Analog Input

In This Chapter. . . .

Module Specifications

The following table provides the specifications for the F3–16AD Analog Input Module from FACTS Engineering. Review these specifications to make sure the module meets your application requirements.

Number of Channels	16, single ended (one common)
Input Ranges	± 5V, ± 10V, 0–5V[1], 0–10V, 0–20 mA, 4 – 20 mA[2]
Resolution	12 bit (1 in 4096)
Input Impedance	2MΩ, voltage input 500Ω ± 1%, current input
Absolute Maximum Ratings	± 25V, voltage input ± 30 mA, current input
Conversion Time	35μs per channel 1 channel per CPU scan
Converter Type	Successive Approximation, AD574
Linearity Error	± 1 count maximum
Maximum Inaccuracy at 77 °F (25 °C)	0.25% of full scale, voltage input 1.25% of full scale, current input
Accuracy vs. Temperature	57 ppm / °C maximum full scale
Recommended Fuse	0.032 A, Series 217 fast-acting, current inputs
Power Budget Requirement	33 mA @ 9 VDC, 47 mA @ 24 VDC
External Power Supply	None required
Operating Temperature	32° to 140° F (0° to 60° C)
Storage Temperature	–4° to 158° F (–20° to 70° C)
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
1 – requires gain adjustment with potentiometer. 2 – resolution is 3275 counts (instead of 4096). Allows easier broken transmitter detection	

Analog Input Configuration Requirements

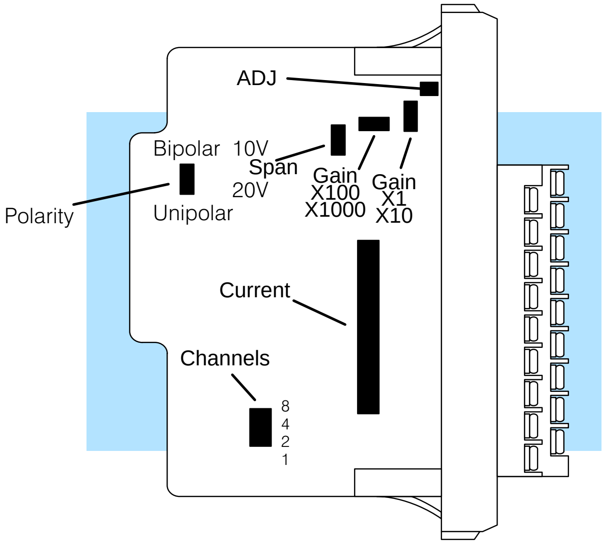
The F3–16AD Analog Input appears as a 16-point module. The module can be installed in any slot configured for 16 points. See the DL305 User Manual for details on using 16 point modules in DL305 systems. The limitation on the number of analog modules are:

- For local and expansion systems, the available power budget and 16-point module usage are the limiting factors.

Setting the Module Jumpers

Jumper Locations

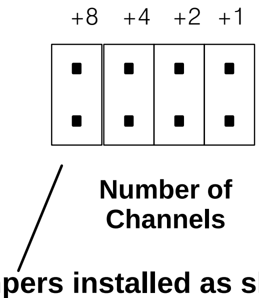
The module is set at the factory for a 0–20 mA signal on all sixteen channels. If this is acceptable you do not have to change any of the jumpers. The following diagram shows the jumper locations.



Selecting the Number of Channels

If you examine the rear of the module, you'll notice several jumpers. The jumpers labeled +1, +2, +4 and +8 are used to select the number of channels that will be used. Without any jumpers the module processes one channel. By installing the jumpers you can add channels. The module is set from the factory for sixteen channel operation.

Any unused channels are not processed so if you only select channels 1–8, then the last eight channels will not be active. The following table shows which jumpers to install.



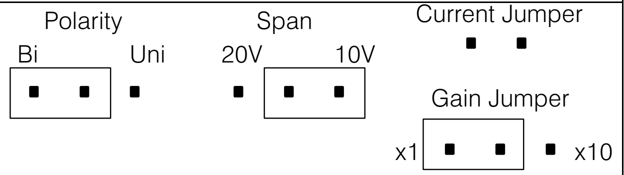
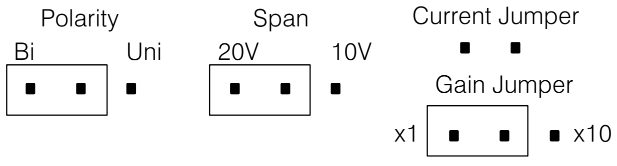
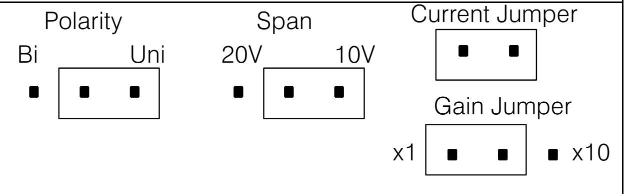
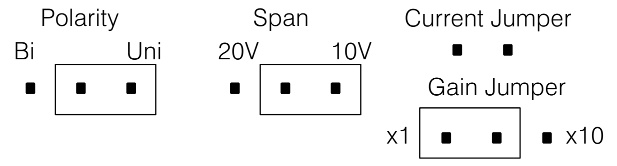
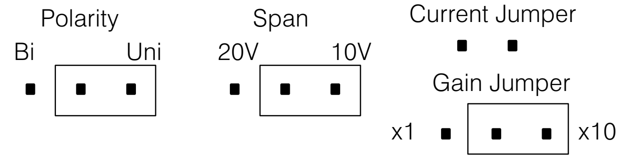
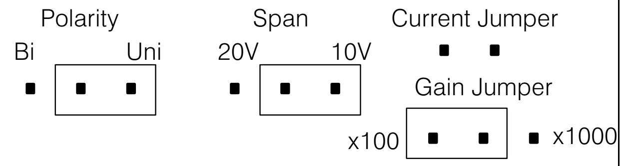
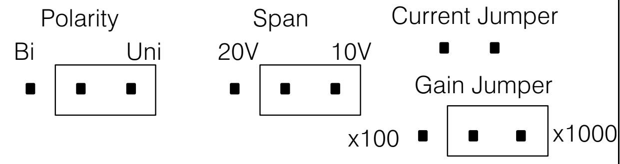
Jumpers installed as shown selects 16-channel operation

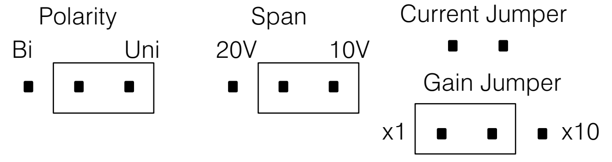
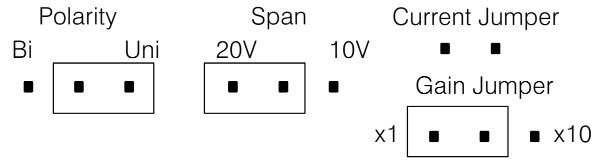
Channel(s)	Jumper			
	+8	**+4**	**+2**	**+1**
1	No	No	No	No
1 2	No	No	No	Yes
1 2 3	No	No	Yes	No
1 2 3 4	No	No	Yes	Yes
1 2 3 4 5	No	Yes	No	No
1 2 3 4 5 6	No	Yes	No	Yes
1 2 3 4 5 6 7	No	Yes	Yes	No
1 2 3 4 5 6 7 8	No	Yes	Yes	Yes
1 2 3 4 5 6 7 8 9	Yes	No	No	No
1 2 3 4 5 6 7 8 9 10	Yes	No	No	Yes
1 2 3 4 5 6 7 8 9 10 11	Yes	No	Yes	No
1 2 3 4 5 6 7 8 9 10 11 12	Yes	No	Yes	Yes
1 2 3 4 5 6 7 8 9 10 11 12 13	Yes	Yes	No	No
1 2 3 4 5 6 7 8 9 10 11 12 13 14	Yes	Yes	No	Yes
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Yes	Yes	Yes	No
1 2 3 4 5 6 7 8 9 10 11 12 13141516	Yes	Yes	Yes	Yes

Selecting Input Signal Ranges

As you examined the jumper settings, you may have noticed there are current jumpers for each individual channel. These jumpers allow you to select the type of signal (voltage or current).

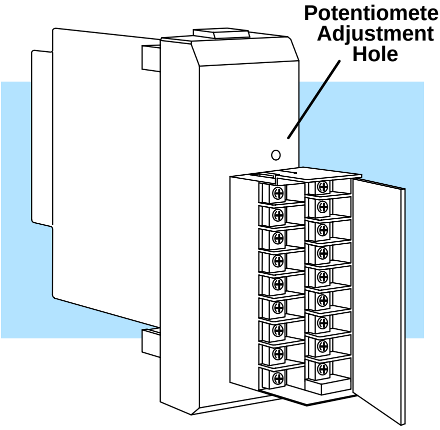
The span and polarity jumpers are used to select the signal range. The polarity and span selection affect all the channels. For example, if you select unipolar operation and a 10V span, you can use both 0 –10V and 0–20 mA signals at the same time. Channels that will receive 0–20 mA signals should have the current jumper installed. The following table shows the jumper selections for the various ranges. (Only channel 1 is used in the example, but all channels must be set.)

Bipolar Signal Range	Jumper Settings
–5 VDC to +5 VDC	
–10 VDC to +10 VDC	
Unipolar Signal Range	**Jumper Settings**
0 to 20 mA (these settings are also used for the 4–20mA range)	
0 VDC to +10 VDC	
0 VDC to +1 VDC	
0 VDC to +0.1 VDC	
0 VDC to +0.01 VDC	

Input Signal Range	Jumper Settings
0 VDC to +5 VDC (requires gain adjustment see instructions below)	
0 VDC to +12 VDC (requires gain adjustment see instructions below)	

Variable Gain Adjustment

If you look at the terminal block closely, you'll notice a small hole conceals an adjustment potentiometer. This small potentiometer is used to adjust the gain for certain situations.



For example, if you have 0–5V transmitters you have to use the 0–10V scale on the module. Since the module converts the signal to a digital value between 0 and 4095, a 5V signal would only yield a value of 2048. Fortunately, the variable gain feature provides a simple solution. Just complete the following steps.

1. Install a jumper on the gain adjustment pins. (This jumper location is labeled ADJ. This jumper will remain installed after the gain adjustment .)
2. Apply 5V to one of the channels.
3. Use a handheld programmer or ***Direct***SOFT to monitor the input register that contains the analog data. (If you're not familiar with this procedure, wait until you read the section on Writing the Control Program. This will show you how to get data into a register. You can come back to this procedure later.)
4. Adjust the potentiometer until the register value reads 4094 or 4095. The potentiometer is turned clockwise to increase the gain.

Now the module has been adjusted so a 5V signal provides a digital value of 4095 instead of 2048.

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.

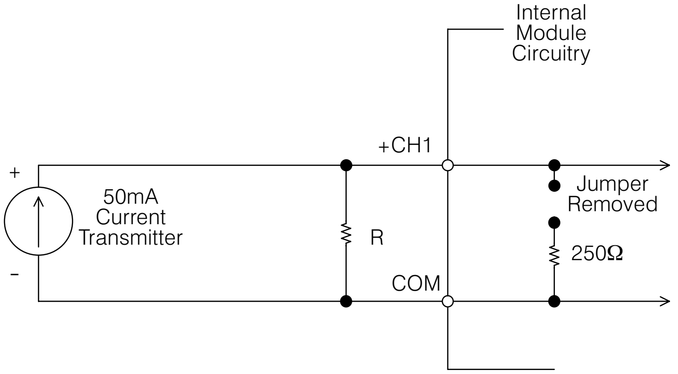
- Use the shortest wiring route whenever possible.
- Use shielded wiring and ground the shield at the signal source. *Do not* ground the shield at both the module and the source.
- Don't run the signal wiring next to large motors, high current switches, or transformers. This may cause noise problems.
- 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 F3–16AD receives all power from the base. A separate power supply is not required.

Custom Input Ranges

Occasionally you may have the need to connect a transmitter with an unusual signal range. By changing the wiring slightly and adding an external resistor to convert the current to voltage, you can easily adapt this module to meet the specifications for a transmitter that does not adhere to one of the standard input ranges. The following diagram shows how this works.



$$R = \frac{V_{max}}{I_{max}}$$

R = value of external resistor
V_{max} = high limit of selected voltage range
I_{max} = maximum current supplied by the transmitter

Example: current transmitter capable of 50mA, 0 – 10V range selected.

$$R = \frac{10V}{50mA} \qquad R = 200 \text{ ohms}$$

NOTE: Your choice of resistor can affect the accuracy of the module. A resistor that has ±0.1% tolerance and a ±50ppm / °C temperature coefficient is recommended.

Current Loop Transmitter Impedance

Standard 4 to 20 mA transmitters and transducers can operate from a wide variety of power supplies. Not all transmitters are alike and the manufacturers often specify a minimum loop or load resistance that must be used with the transmitter at the various voltages.

The F3–16AD provides 500 ohm resistance for each channel. If your transmitter requires a load resistance below 500 ohms, then you do not have to make any adjustments. However, if your transmitter requires a load resistance higher than 500 ohms, then you need to add a resistor in series with the module.

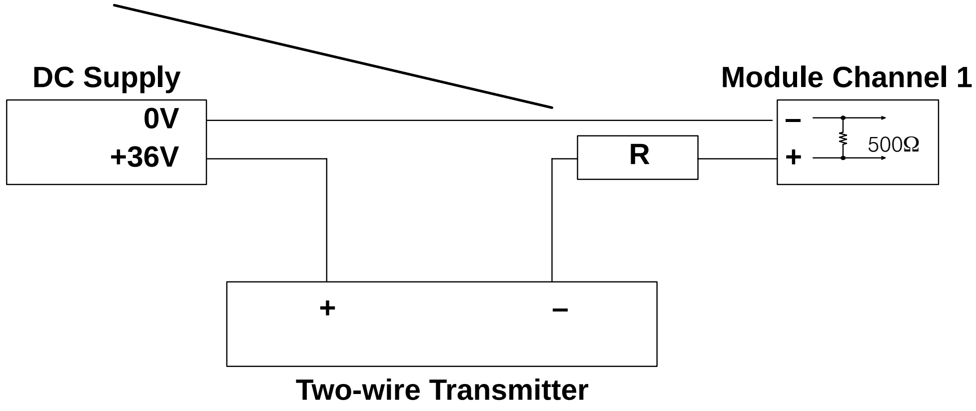
Consider the following example for a transmitter being operated from a 36 VDC supply with a recommended load resistance of 750 ohms. Since the module has a 500 ohm resistor, you need to add an additional resistor.

$R = Tr - Mr$

$R = 750 - 500$

$R \geq 250$

R – Resistor to add
Tr – Transmitter Requirement
Mr – Module resistance (internal 500 ohms)



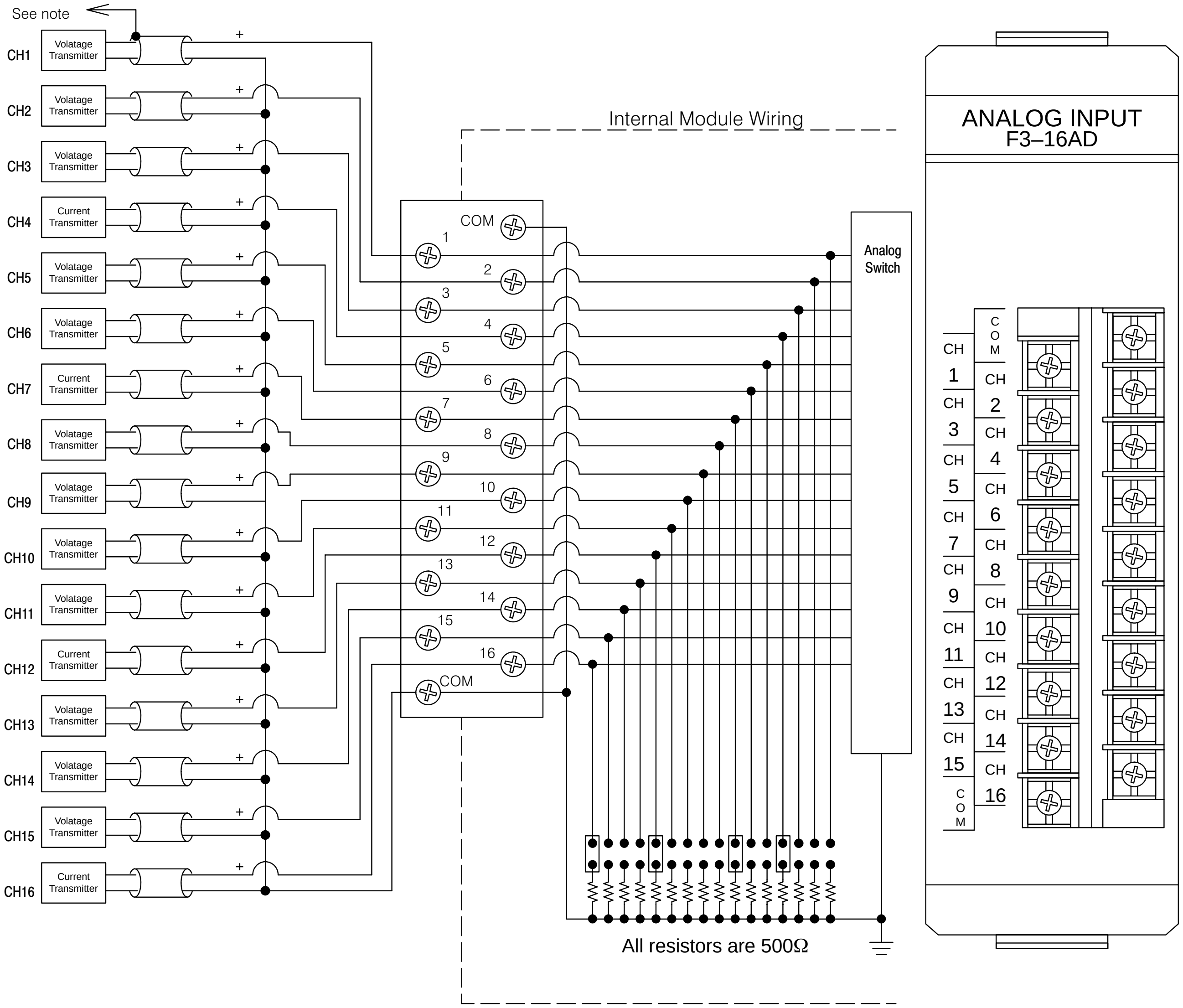
Removable Connector

The F3–16AD module has a removable connector to make wiring easier. Simply squeeze the top and bottom tabs and gently pull the connector from the module.

See Errata Sheet at the beginning of this file. The wiring diagram has been revised to show 2-wire and 4-wire current transmitter examples with connections to external 24 VDC power supplies.

Wiring Diagram

Note 1: Terminate all shields at their respective signal source.
Note 2: Jumpers for CH4, 7, 12 and 16 are installed for current input.



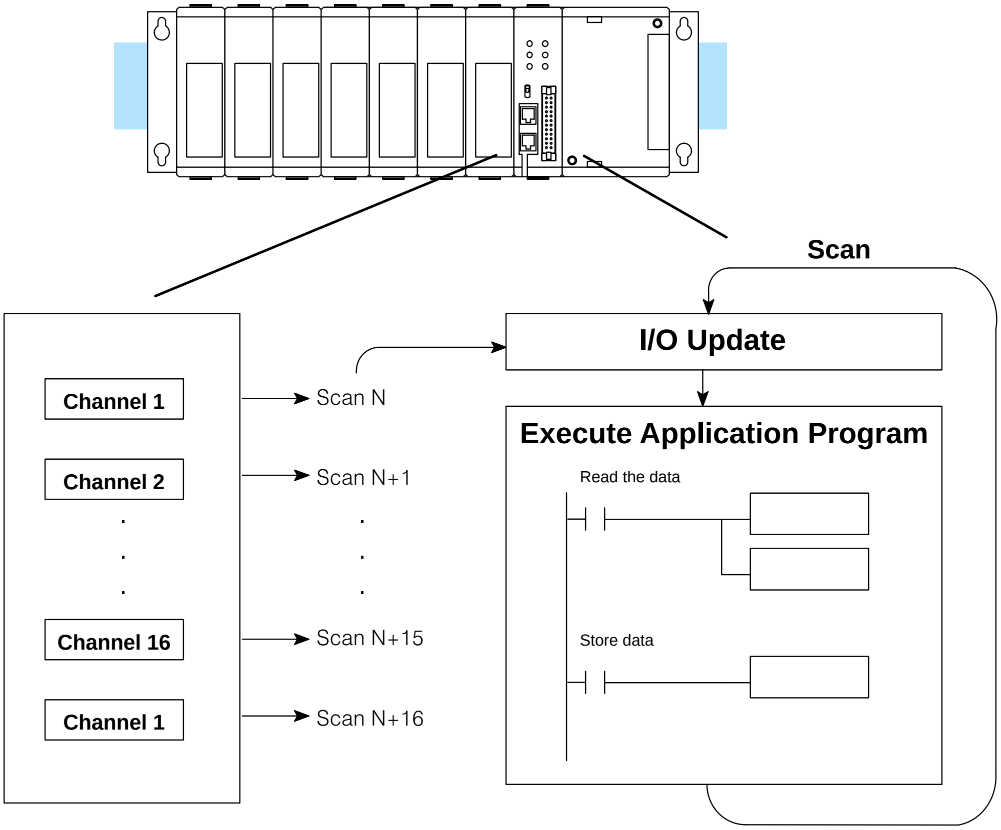
Module Operation

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 F3–16AD module supplies 1 channel of data per each CPU scan. Since there are sixteen channels, it can take up to sixteen scans to get data for all channels. Once all channels have been scanned the process starts over with channel 1.

You do not have to select all of the channels. Unused channels are not processed, so if you select only eight channels, then the channels will be updated within eight scans.



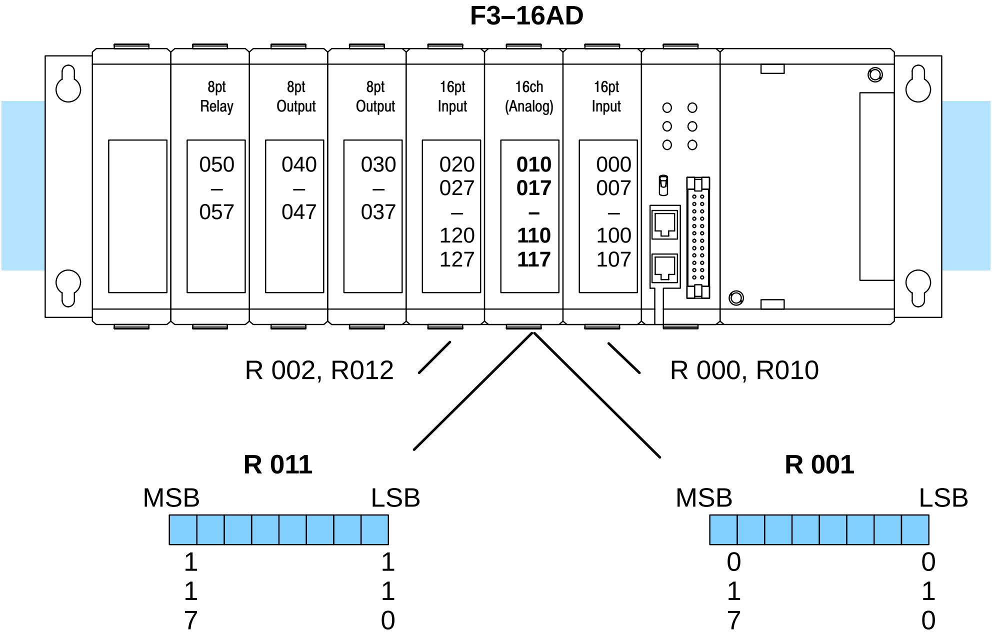
Even though the channel updates to the CPU are synchronous with the CPU scan, the module asynchronously monitors the analog transmitter signal and converts the signal to a 12-bit binary representation. This enables the module to continuously provide accurate measurements without slowing down the discrete control logic in the RLL program.

Understanding the I/O Assignments

You may recall the F3–16AD module appears to the CPU as a 16-point module. These 16 points provide:

- an indication of which channel is active.
- the digital representation of the analog signal.

Since all I/O points are automatically mapped into Register (R) memory, it is very easy to determine the location of the data word that will be assigned to the module.

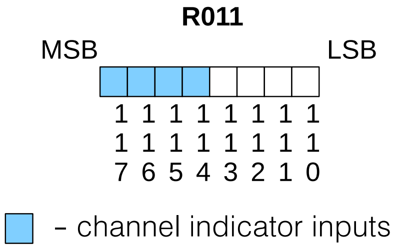


Within these two register locations, the individual bits represent specific information about the analog signal.

Active Channel Indicator Inputs

The last four inputs of the upper Register indicate the active channel. The indicators automatically increment with each CPU scan.

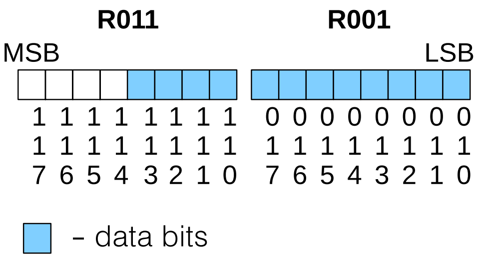
Scan	Channel Inputs	Active Channel
N	0000	1
N+1	0001	2
N+2	0010	3
N+3	0011	4
N+4	0100	5
N+5	0101	6
N+6	0110	7
N+7	0111	8
N+8	1000	9
N+9	1001	10
N+10	1010	11
N+11	1011	12
N+12	1100	13
N+13	1101	14
N+14	1110	15
N+15	1111	16



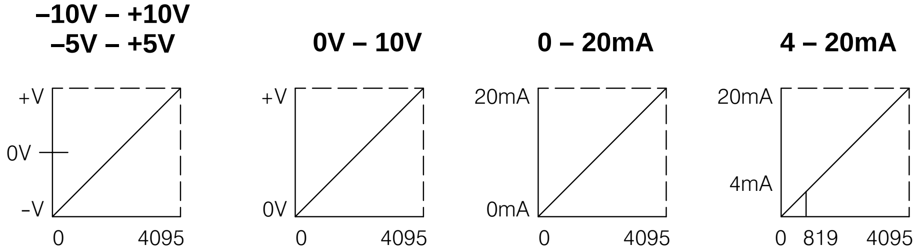
Analog Data Bits

The remaining twelve bits represent the analog data in binary format.

Bit	Value	Bit	Value
0 (LSB)	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 converted into 4096 “pieces” ranging from 0 – 4095 (2^{12}). For example, with a 0 to 10V scale, a 0V signal would be 0, and a 10V signal would be 4095. This is equivalent to a 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.



NOTE: When you use 4–20mA signals, you have to use the 0–20mA scale. You do not have resolution of 4096 if the 4–20mA signal is present. In this case, the range is 819 to 4095. This is because a 0 still represents 0mA, not 4mA.

Each “piece” can also be expressed in terms of the signal level by using the equation shown. The following table shows the smallest signal levels that will possibly result in a change in the data value for each signal range.

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

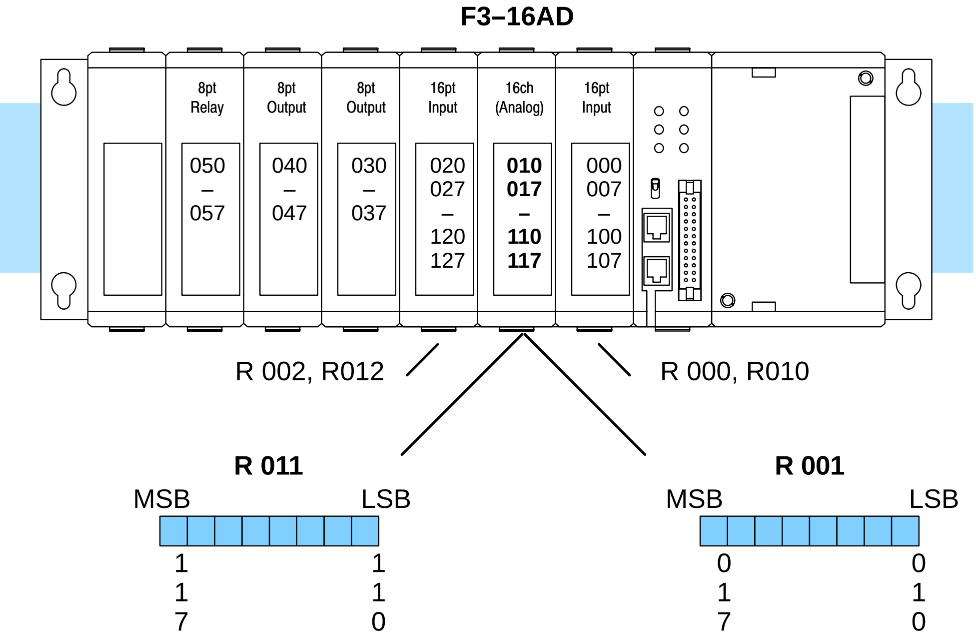
H = high limit of the signal range
L = low limit of the signal range

Range	Highest Signal	Lowest Signal	Smallest Change
–10 to +10V	+10V	–10V	4.88 mV
–5 to +5V	+5 V	–5V	2.44 mV
0 to 5V	5V	0V	1.22 mV
0 to 10V	10V	0V	2.44 mV
0 to 12V	12V	0V	2.90 mV
0 to 20mA (4 to 20mA also)	20mA	0mA	4.88 μA
0 to 1V	1 V	0 V	0.244 mV
0 to 0.1V	0.1 V	0 V	24.4 uV
0 to 0.01V	0.01 V	0 V	2.44 uV

Writing the Control Program (DL330 / DL340)

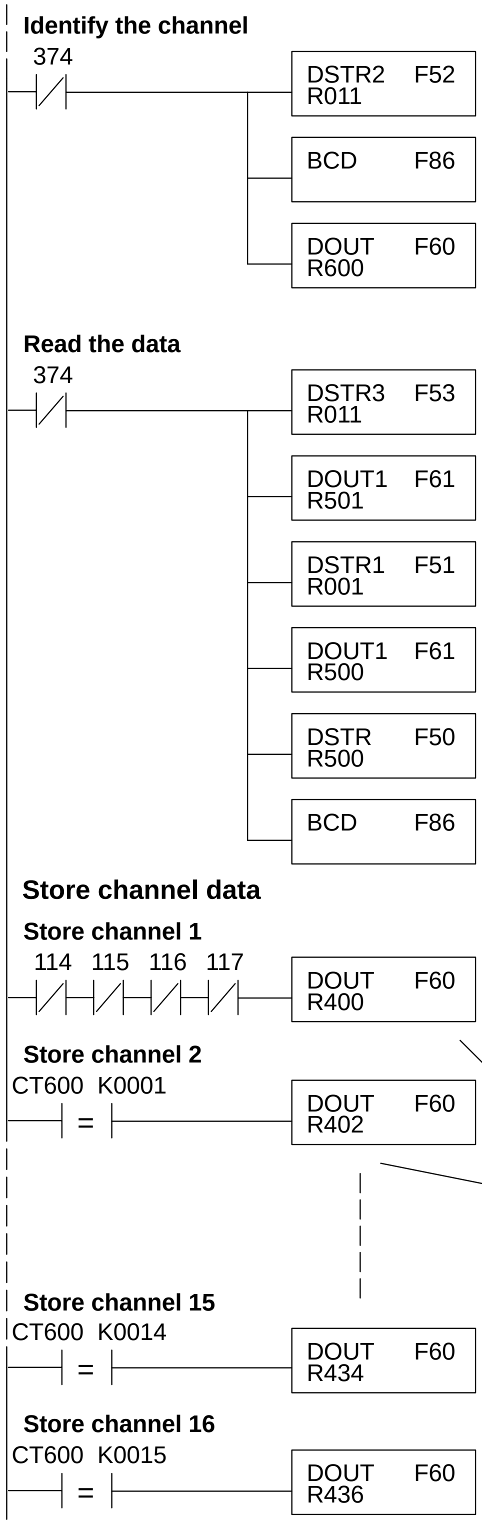
Identifying the Data Locations

Since all channels are multiplexed into a single data word, the control program must be setup to determine which channel is being read. Since the module provides input points to the CPU, it is very easy to use the active channel status bits to determine which channel is being monitored.



Example Program

The following example shows a program designed to read any of the available channels of analog data into Register locations. Once the data is in a Register, you can perform math on the data, compare the data against preset values, etc. Since the DL305 CPUs use 8-bit word instructions, you have to move the data in pieces. It's pretty simple if you follow the example.



This rung loads the channel ID bits into the accumulator from Register 011 on every scan.

Convert the channel ID status to BCD. (We'll use relational contacts later to make the chanel selection much easier.)

Store the channel ID in R600. (Note, you don't absolutely have to do it this way. If you use R600, then you can't use Timer/Counter 600. You could just use the channel indicators. See the Store Channel 1 example that follows.)

This rung loads the four least significant data bits into the accumulator from Register 011 on every scan.

Temporarily store the bits to Register 501.

This rung loads the eight least significant data bits into the accumulator from Register 001.

Temporarily store the bits to Register 500. Since the most significant bits were loaded into 501, now R500 and R501 contain all twelve bits in order.

Now that all the bits are stored, load all twelve bits into the accumulator.

Math operations are performed in BCD. This instruction converts the binary data to BCD. (You can omit this step if your application does not require the conversion.)

The channel selection inputs are used to let the CPU know which channel has been loaded into the accumulator. By using these inputs to control a DOUT instruction, you can easily move the data to a storage register. Notice the DOUT instruction stores the data in two bytes. (Two bytes are required for four digit BCD numbers.)

This rung shows how you would use the channel indicator inputs as contacts to control the channel selection.

This rung shows an easier way. Earlier we loaded the channel ID bits (in BCD format) into R600. Now we can use one relational contact to examine this value. However, this method uses the register associated with Timer/Counter 600. If you use this method, make sure you don't use the Timer/Counter associated with the register elsewhere in the program.

Scaling the Input Data

Most applications usually require measurements in engineering units, which provide more meaningful data. This is accomplished by using the conversion formula shown.

$$\text{Units} = \frac{A}{4096}\,S$$

Units = value in Engineering Units

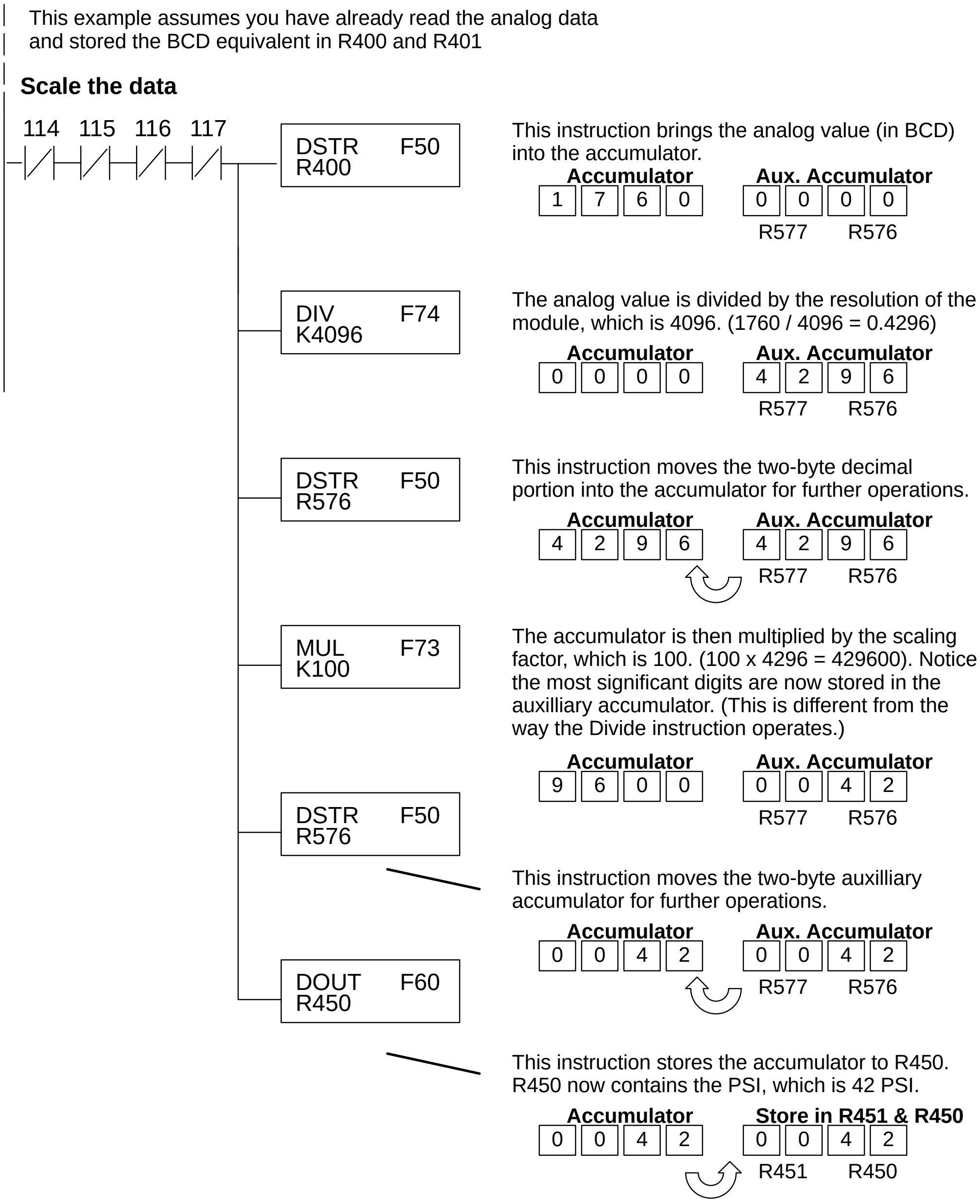
A = Analog value (0 – 4095)

S = high limit of the Engineering unit range

The following example shows how you would use the analog data to represent pressure (PSI) from 0 to 100. This example assumes the analog value is 1760. This should yield approximately 42.9 PSI.

$$\begin{tikzcd}
\text{Units} = \frac{A}{4096}\,S \arrow[d] \\
\text{Units} = \frac{1760}{4096}\,100 \arrow[d] \\
\text{Units} = 42.9
\end{tikzcd}$$

The following instructions are required to scale the data. (We'll continue to use the 42.9 PSI example.) In this example we're using channel 1. The active channel indicator inputs are all off when channel 1 data is being read. Of course, if you were using a different channel, you would use the active channel indicator point combination that corresponds to the channel you were using.



This instruction brings the analog value (in BCD) into the accumulator.

The analog value is divided by the resolution of the module, which is 4096. (1760 / 4096 = 0.4296)

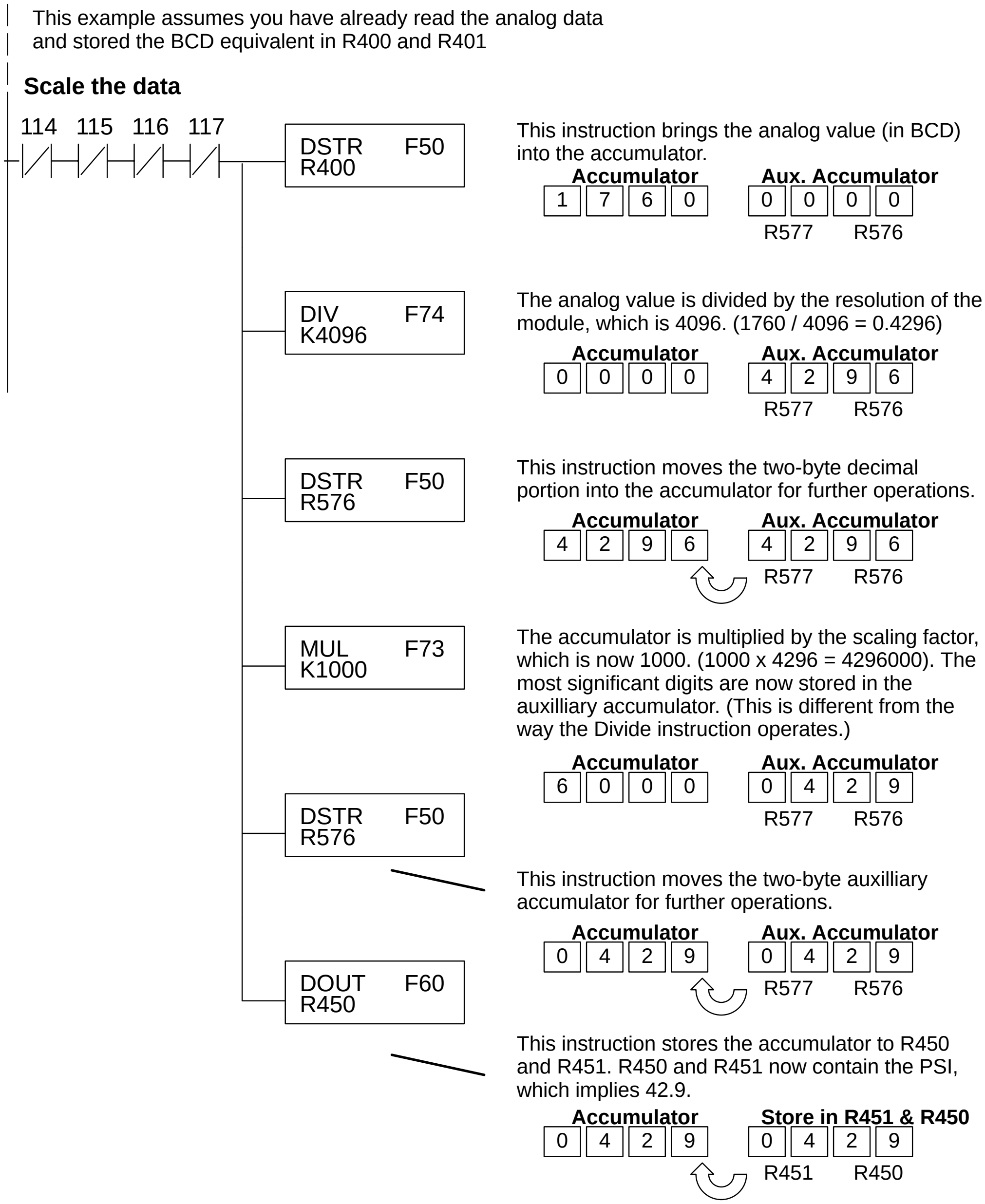
This instruction moves the two-byte decimal portion into the accumulator for further operations.

The accumulator is then multiplied by the scaling factor, which is 100. (100 x 4296 = 429600). Notice the most significant digits are now stored in the auxilliary accumulator. (This is different from the way the Divide instruction operates.)

This instruction moves the two-byte auxilliary accumulator for further operations.

This instruction stores the accumulator to R450. R450 now contains the PSI, which is 42 PSI.

You probably noticed the previous example yielded 42 PSI when the real value should have been 42.9 PSI. By changing the scaling value slightly, we can "imply" an extra decimal of precision. Notice in the following example we've added another digit to the scale. Instead of a scale of 100, we're using 1000, which implies 100.0 for the PSI range.



This instruction brings the analog value (in BCD) into the accumulator.

Accumulator	Aux. Accumulator
1 7 6 0	0 0 0 0
	R577 R576

The analog value is divided by the resolution of the module, which is 4096. (1760 / 4096 = 0.4296)

Accumulator	Aux. Accumulator
0 0 0 0	4 2 9 6
	R577 R576

This instruction moves the two-byte decimal portion into the accumulator for further operations.

Accumulator	Aux. Accumulator
4 2 9 6	4 2 9 6
	R577 R576

The accumulator is multiplied by the scaling factor, which is now 1000. (1000 x 4296 = 4296000). The most significant digits are now stored in the auxilliary accumulator. (This is different from the way the Divide instruction operates.)

Accumulator	Aux. Accumulator
6 0 0 0	0 4 2 9
	R577 R576

This instruction moves the two-byte auxilliary accumulator for further operations.

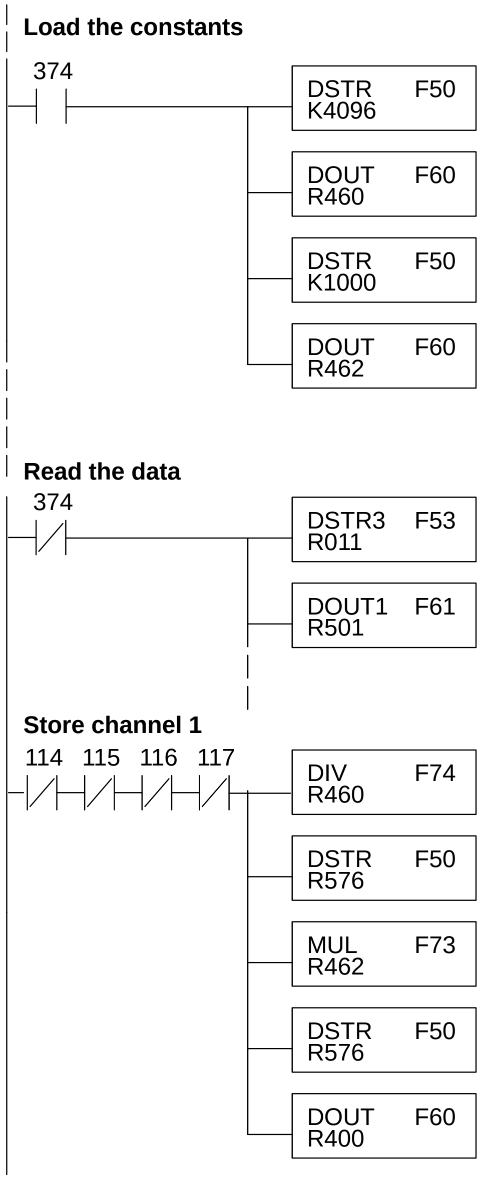
Accumulator	Aux. Accumulator
0 4 2 9	0 4 2 9
	R577 R576

This instruction stores the accumulator to R450 and R451. R450 and R451 now contain the PSI, which implies 42.9.

Accumulator	Store in R451 & R450
0 4 2 9	0 4 2 9
	R451 R450

This example program shows how you can use the instructions to load these equation constants into data registers. The example is written for channel 1, but you can easily use a similar approach to use different scales for all channels if required.

You may just use the appropriate constants in the instructions dedicated for each channel, but this method allows easier modifications. For example, you could easily use an operator interface or a programming device to change the constants if they are stored in Registers.



On the first scan, these first two instructions load the analog resolution (constant of 4096) into R460 and R461.

These two instructions load the high limit of the Engineering unit scale (constant of 1000) into R462 and R463. Note, if you have different scales for each channel, you'll also have to enter the Engineering unit high limit for those as well.

This rung loads the four most significant data bits into the accumulator from Register 011 on every scan.

Temporarily store the bits to Register 501.

The analog value is divided by the resolution of the module, which is stored in R460.

This instruction moves the decimal portion from the auxilliary accumulator into the regular accumulator for further operations.

The accumulator is multiplied by the scaling factor, which is stored in R462.

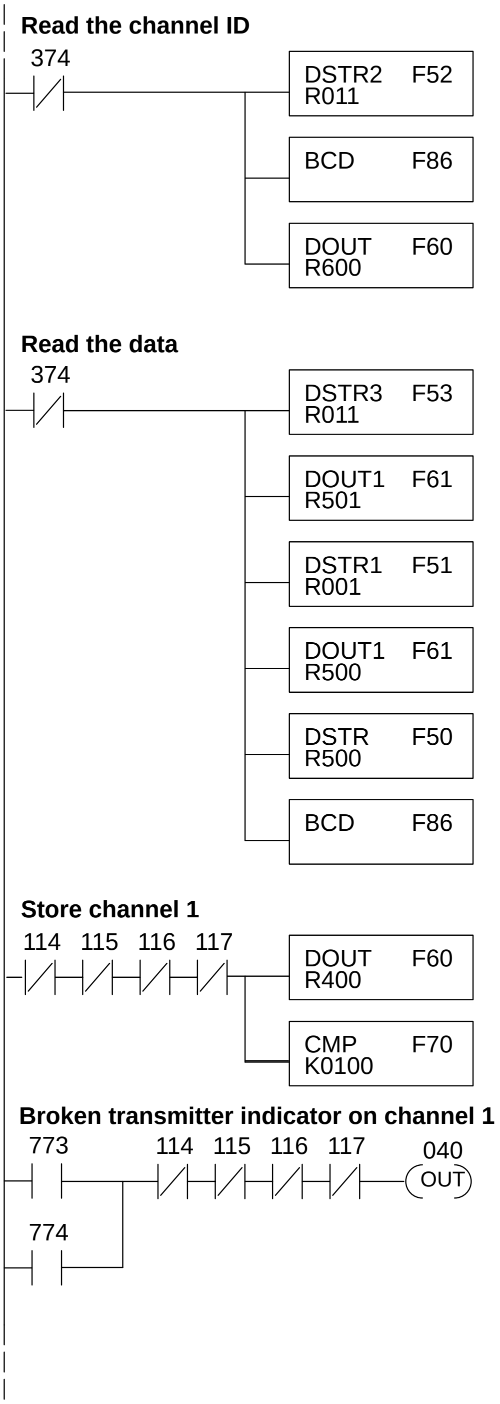
This instruction moves most significant digits (now stored in the auxilliary accumulator) into the regular accumulator for further operations.

The scaled value is stored in R400 and R401 for further use.

Broken Transmitter Detection

If you use 4–20mA signals you can easily check for broken transmitter conditions. Since you have to use the 0–20mA range and the lowest signal for the 4–20mA transmitter is 4mA, the lowest digital value for the signal is not 0, but instead is 819.

If the transmitter is working properly the smallest value you should ever see is 819. If you see a value of less than about 750 (allowing for tolerance), then you know the transmitter is broken.



This rung loads the channel ID bits into the accumulator from Register 011 on every scan.

Convert the channel ID status to BCD. We'll use relational contacts later to make the chanel selection much easier.)

Store the channel ID in R600.

This rung loads the four most significant data bits into the accumulator from Register 011 on every scan.

Temporarily store the bits to Register 501.

This rung loads the eight least significant data bits into the accumulator from Register 001.

Temporarily store the bits to Register 500. Since the most significant bits were loaded into 501, now R500 and R501 contain all twelve bits in order.

Now that all the bits are stored, load all twelve bits into the accumulator.

Math operations are performed in BCD. This instruction converts the binary data to BCD. (You can omit this step if your application does not require the conversion.)

The DOUT instruction copies the accumulator data to R400 and R401. Since the data is still in the accumulator, we can compare it against a constant. Since the minimum value for a 4mA signal is 819 (minus the module tolerance), we can choose a value for the compare. We picked 100, but you could choose something else from 0 to about 750.

Flags 773 and 774 are used with the Compare instruction. In this example if the analog value is less than or equal to 100, then output 040 is turned on.

You may want to latch 040 to catch intermittent broken transmitters.

Writing the Control Program (DL350)

Reading Values: Pointer Method and Multiplexing

There are two methods of reading values for the DL350:

- The pointer method **(all system bases must be D3–xx–1 bases to support the pointer method)**
- Multiplexing

You must use the multiplexing method with remote I/O modules (the pointer method will not work). You can use either method when using DL350, but for ease of programming it is strongly recommended that you use the pointer method.

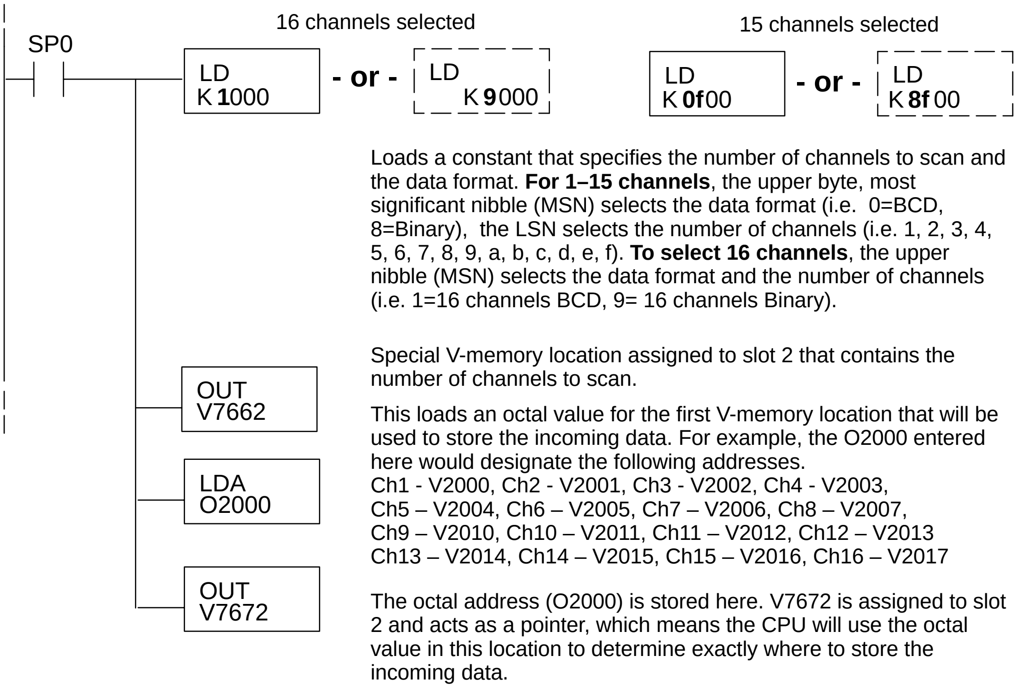
NOTE: Do not use the pointer method and the PID PV auto transfer from I/O module function together for the same module. If using PID loops, use the pointer method and ladder logic code to map the analog input data into the PID loop table.

Pointer Method

The DL350 has special V-memory locations assigned to each base slot that greatly simplifies the programming requirements. These V-memory locations allow you to:

- specify the data format
- specify the number of channels to scan
- specify the storage locations

The example program shows how to setup these locations. Place this rung anywhere in the ladder program or in the Initial Stage if you are using RLL*PLUS* instructions. This is all that is required to read the data into V-memory locations. Once the data is in V-memory, you can perform math on the data, compare the data against preset values, and so forth. V2000 is used in the example, but you can use any user V-memory location. In this example the module is installed in slot 2. You should use the V-memory locations for your module placement.



Loads a constant that specifies the number of channels to scan and the data format. **For 1–15 channels**, the upper byte, most significant nibble (MSN) selects the data format (i.e. 0=BCD, 8=Binary), the LSN selects the number of channels (i.e. 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f). **To select 16 channels**, the upper nibble (MSN) selects the data format and the number of channels (i.e. 1=16 channels BCD, 9= 16 channels Binary).

Special V-memory location assigned to slot 2 that contains the number of channels to scan.

This loads an octal value for the first V-memory location that will be used to store the incoming data. For example, the O2000 entered here would designate the following addresses.
Ch1 - V2000, Ch2 - V2001, Ch3 - V2002, Ch4 - V2003,
Ch5 – V2004, Ch6 – V2005, Ch7 – V2006, Ch8 – V2007,
Ch9 – V2010, Ch10 – V2011, Ch11 – V2012, Ch12 – V2013
Ch13 – V2014, Ch14 – V2015, Ch15 – V2016, Ch16 – V2017

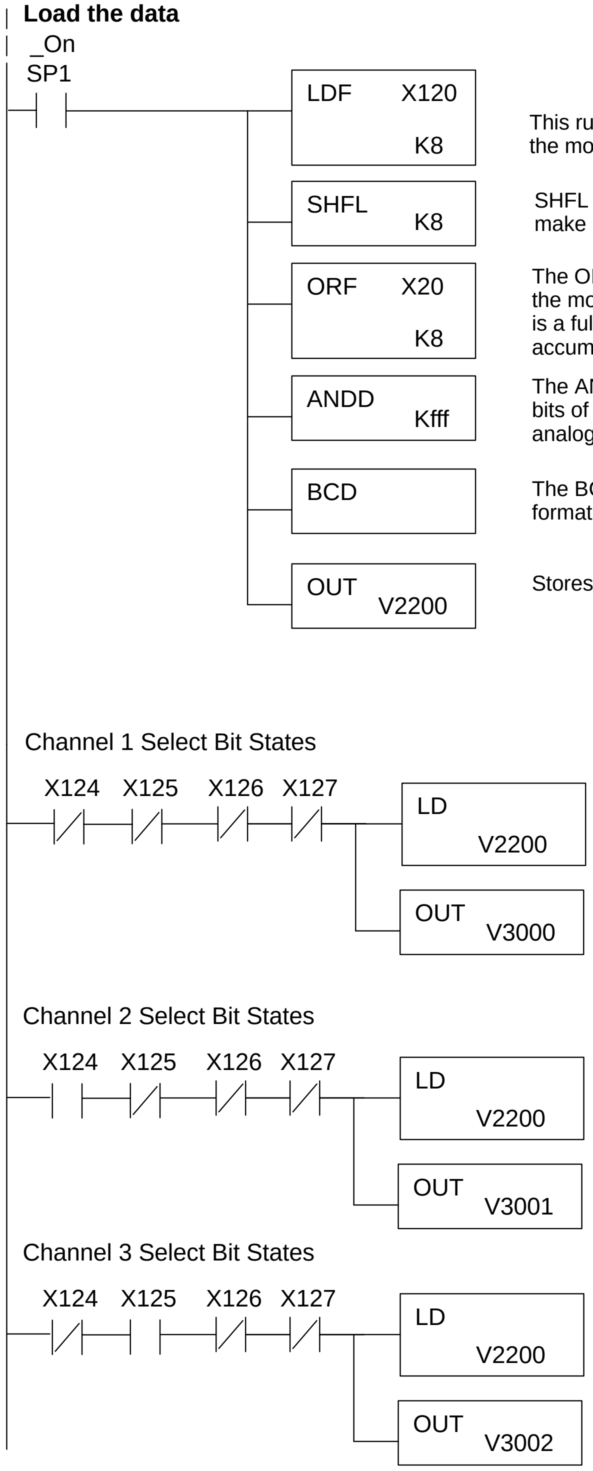
The octal address (O2000) is stored here. V7672 is assigned to slot 2 and acts as a pointer, which means the CPU will use the octal value in this location to determine exactly where to store the incoming data.

The table shows the special V-memory locations used with the DL350. Slot 0 (zero) is the module next to the CPU, slot 1 is the module two places from the CPU, and so on. Remember, the CPU only examines the pointer values at these locations after a mode transition. The pointer method is supported on expansion bases up to a total of 8 slots away from the DL350 CPU. The pointer method is not supported in slot 8 of a 10 slot base.

Analog Input Module Slot-Dependent V-memory Locations								
Slot	0	1	2	3	4	5	6	7
No. of Channels	V7660	V7661	V7662	V7663	V7664	V7665	V7666	V7667
Storage Pointer	V7670	V7671	V7672	V7673	V7674	V7675	V7676	V7677

Multiplexing: DL350 with a Conventional DL305 Base

The example below shows how to read multiple channels on an F3–08AD Analog module in the 20–27/120–127 address slot. This module must be placed in a 16 bit slot in order to work.



This rung loads the upper byte of analog data from the module.

SHFL K8 shifts the data to the left eight places to make room for the lower byte of data.

The ORF X20 brings the lower byte of data from the module into the accumulator. At this time there is a full word of data from the analog module in the accumulator.

The ANDD Kfff masks off the four most significant bits of data from the word. This leaves the actual analog value.

The BCD command converts the data to BCD format.

Stores the data in V2200.

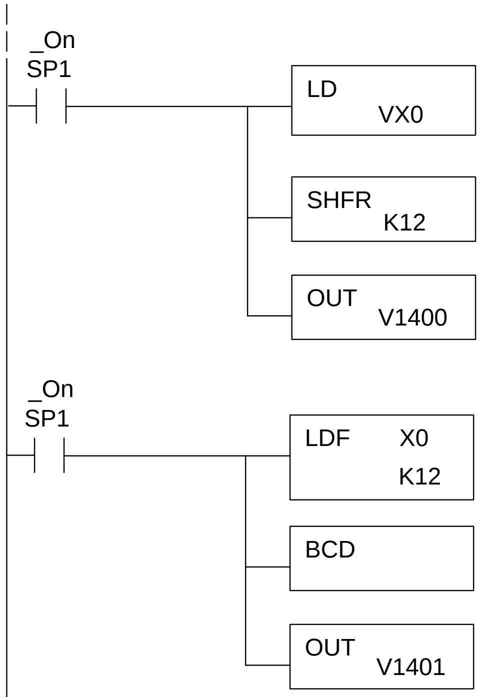
This sends channel one analog data to V3000 when bits X124, X125, X126 and X127 are as shown.

This sends channel two analog data to V3001 when bits X124, X125, X126 and X127 are as shown.

This sends channel two analog data to V3002 when bits X124, X125, X126 and X127 are as shown.

Multiplexing: DL350 with a D3–XX–1 Base

The example below shows how to read multiple channels on an F3–16AD Analog module in the X0 address slot of the D3–XX–1 base. If any expansion bases are used in the system, they must all be D3–xx–1 to be able to use this example. Otherwise, the conventional base addressing must be used.



This rung loads the upper byte of analog data from the module.

SHFL K12 shifts the word to the right twelve places.

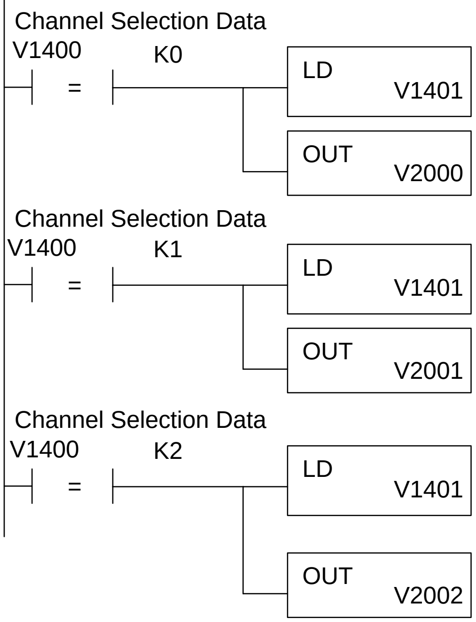
Puts the four channel select bits in the lower nibble (four bits) of word V1400. This will increment once with each scan from 0 to F.

This rung loads the twelve bits of analog data to the module and converts it to BCD. It is the OUT to V1401.

This converts the data to BCD.

The analog data (in BCD format) is then stored in the Holding Register, V1401.

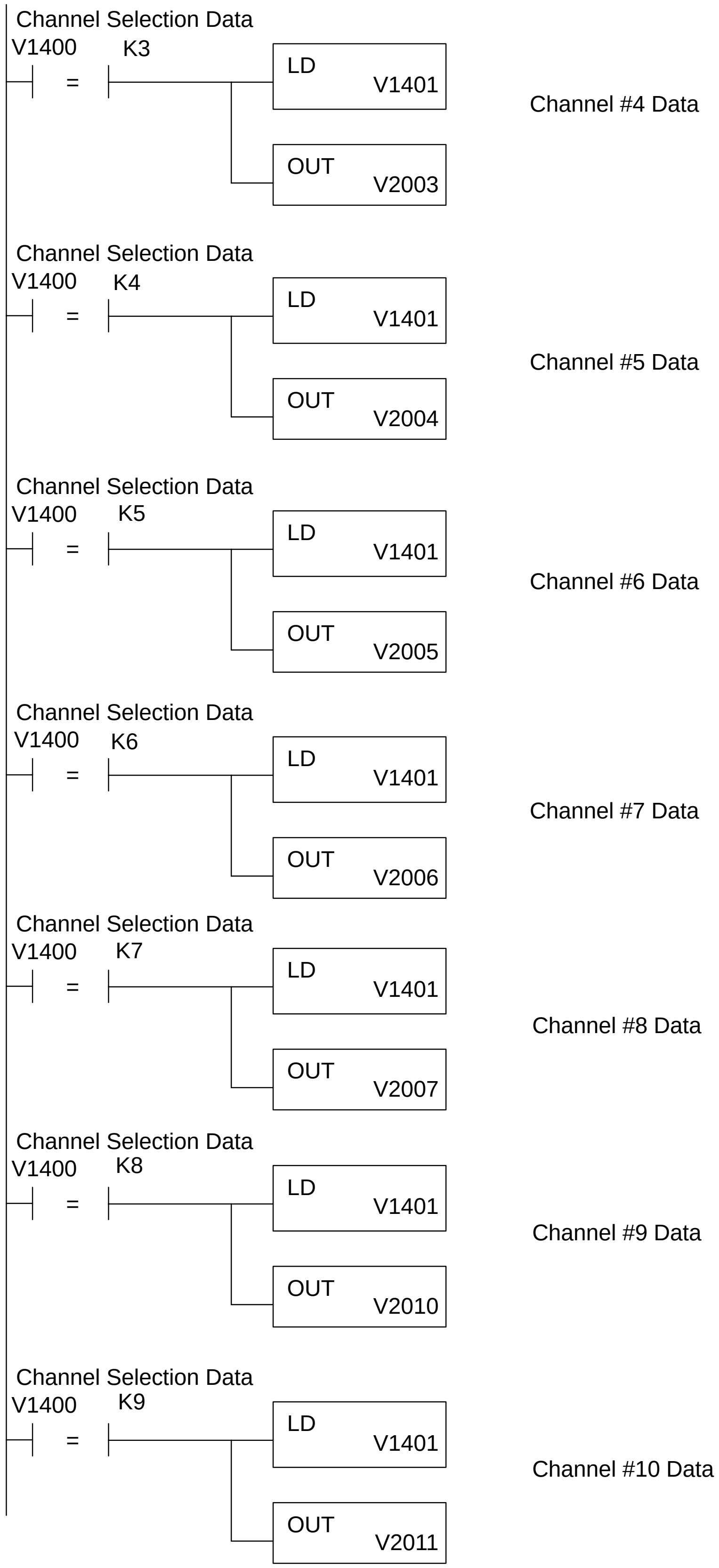
Rungs 3–18 compare the count of the chennel select bits. When the corresponding bits are true, the channel data for that channel is stored in the proper V-memory location. For sixteen channels of analog data, the module will require sixteen scans in order to update all channels.

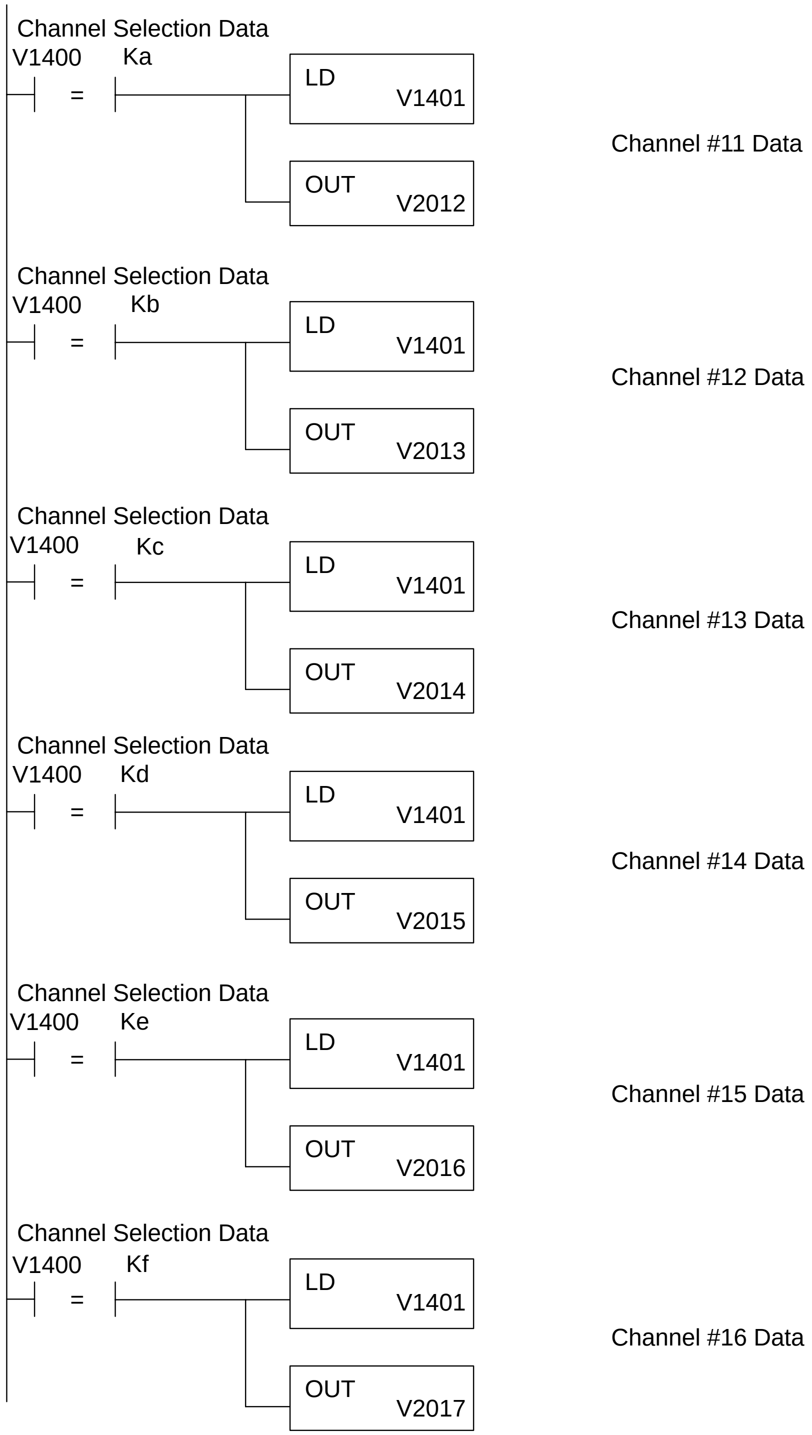


Channel #1 Data

Channel #2 Data

Channel #3 Data


Channel Selection Data
V1400
K3
=
LD
V1401
OUT
V2003
Channel #4 Data
Channel Selection Data
V1400
K4
=
LD
V1401
OUT
V2004
Channel #5 Data
Channel Selection Data
V1400
K5
=
LD
V1401
OUT
V2005
Channel #6 Data
Channel Selection Data
V1400
K6
=
LD
V1401
OUT
V2006
Channel #7 Data
Channel Selection Data
V1400
K7
=
LD
V1401
OUT
V2007
Channel #8 Data
Channel Selection Data
V1400
K8
=
LD
V1401
OUT
V2010
Channel #9 Data
Channel Selection Data
V1400
K9
=
LD
V1401
OUT
V2011
Channel #10 Data


Channel Selection Data
V1400
Ka
=
LD
V1401
OUT
V2012
Channel #11 Data
Channel Selection Data
V1400
Kb
=
LD
V1401
OUT
V2013
Channel #12 Data
Channel Selection Data
V1400
Kc
=
LD
V1401
OUT
V2014
Channel #13 Data
Channel Selection Data
V1400
Kd
=
LD
V1401
OUT
V2015
Channel #14 Data
Channel Selection Data
V1400
Ke
=
LD
V1401
OUT
V2016
Channel #15 Data
Channel Selection Data
V1400
Kf
=
LD
V1401
OUT
V2017
Channel #16 Data

Scaling the Input Data

Most applications usually require measurements in engineering units, which provide more meaningful data. 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.

$$\text{Units} = A \frac{H - L}{4095}$$

H = high limit of the engineering unit range

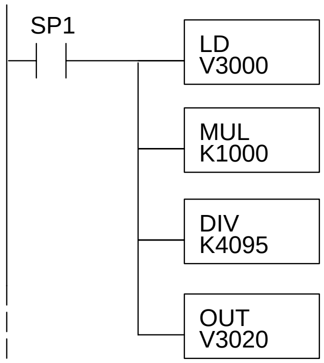
L = low limit of the engineering unit range

A = Analog value (0 – 4095)

For example, if you wanted to measure pressure (PSI) from 0.0 to 99.9 then you would have to multiply the analog value by 10 in order to imply a decimal place when you view the value with the programming software or a handheld programmer. Notice how the calculations differ when you use the multiplier.

Here is how you would write the program to perform the engineering unit conversion. This example assumes you have BCD data loaded into the appropriate V-memory locations using instructions that apply for the model of CPU you are using.

NOTE: This example uses SP1, which is always on. You could also use an X, C, etc. permissive contact.



When SP1 is on, load channel 1 data to the accumulator.

Multiply the accumulator by 1000 (to start the conversion).

Divide the accumulator by 4095.

Store the result in V3020.

Analog and Digital Value Conversions

Sometimes it is helpful to be able to quickly convert between the 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 ...
–10V to + 10V	$A = \frac{20D}{4095} - 10$	$D = \frac{4095}{20}(A + 10)$
–5V to + 5V	$A = \frac{10D}{4095} - 5$	$D = \frac{4095}{10}(A + 5)$
0 to 5V	$A = \frac{5D}{4095}$	$D = \frac{4095}{5} A$
0 to 10V	$A = \frac{10D}{4095}$	$D = \frac{4095}{10} A$
0 to 12V	$A = \frac{12D}{4095}$	$D = \frac{4095}{12} A$
0 to 20mA (or 4–20mA)	$A = \frac{20D}{4095}$	$D = \frac{4095}{20} A$
0 to 1V	$A = \frac{1D}{4095}$	$D = \frac{4095}{1} A$
0 to 0.1V	$A = \frac{0.1D}{4095}$	$D = \frac{4095}{0.1} A$
0 to 0.01V	$\ell A = \frac{0.01D}{4095}$	$D = \frac{4095}{0.01} A$

For example, if you are using the –10 to +10V range and you have measured the signal at 6V, you would use the following formula to determine the digital value that should be stored in the register location that contains the data.

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

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

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

$$D = 3276$$