Application on Control Technology

applications
& TOOLS

Programming Example in Ladder Logic

SIEMENS

Scaling and Unscaling Analog Values

Note

The application examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The application examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible in ensuring that the described products are correctly used. These application examples do not relieve you of the responsibility in safely and professionally using, installing, operating and servicing equipment. When using these application examples, you recognize that Siemens cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these application examples at any time without prior notice. If there are any deviations between the recommendations provided in these application examples and other Siemens publications - e.g. Catalogs - then the contents of the other documents have priority.

Warranty, liability and support

We do not accept any liability for the information contained in this document.

Any claims against us - based on whatever legal reason - resulting from the use of the examples, information, programs, engineering and performance data etc., described in this application example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health The above provisions does not imply a change in the burden of proof to your detriment.

For questions about this document please use the following e-mail address:

mailto:csweb@ad.siemens.de

Table of Contents

Reference to Automation and Drives Service & Support

This entry is from the internet application portal of Automation and Drives Service & Support. The documentation has the entry ID **23330722**. Click the link below to directly display the download page of this document.

http://support.automation.siemens.com/WW/view/en/23330722

All entries referenced in this document are designated by their entry ID and addressed via the above path.

Scaling and Unscaling Analog Values

Overview

This sample program provides an example for using the standard S7 library functions for scaling and unscaling analog values. Included in the program is an FC41 ("Scaling and Unscaling an Analog Value") that calls both the Scale function (FC105) and the Unscale function (FC106) and also provides logic for checking the scaled value against pre-determined limits (out-of-range values).

Note To use FC105 ("SCALE") and FC106 (UNSCALE"), you copy the blocks from the STEP 7 libraries:
Libraries > Standard Library > TI-S7 Converting Blocks.

Objectives

This example provides sample LAD instructions for the following tasks:

- Using the library SCALE Function (FC105) to scale the value for an analog input
- Using cyclic interrupt organization blocks
- Rounding and comparing integer values
- Using the library UNSCALE Function (FC106) to unscale the value for an analog output

Operation of the sample program

The sample program performs the following tasks:

- The S7 controller executes OB31 every 2 seconds (2000 ms), which calls FC41 to perform the scale and unscale operations.
- FC41 calls FC105 ("Scale") to scale an analog value to a Real number (a floating-point value in IEEE notation) between a specified range.
- FC41 also checks that the scaled value is within pre-determined limits by performing the following operations:
  - Converts (rounds) the scaled Real number to a double-integer value
  - Compares the double-integer value to both a high-limit value and a low-limit value
  - Sets a bit to signal if the value is out-of-range
- FC41 then calls FC106 ("Unscale") to convert the Real number to an integer value between a specified range.

1 Using FC41 to scale and unscale analog values

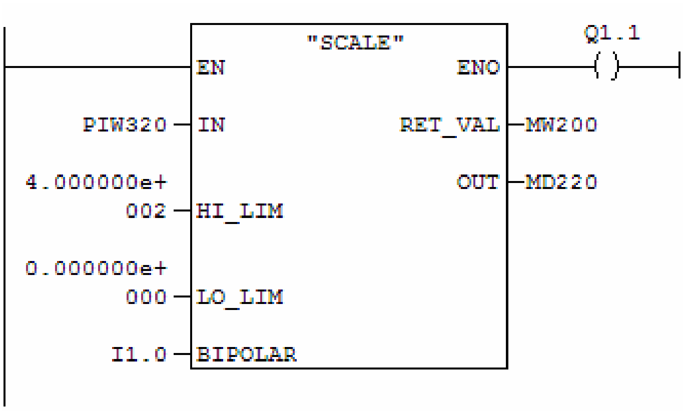
FC41 ("Scaling and Unscaling an Analog Value") provides sample logic for using the Scale function (FC105) and the Unscale function (FC106).

1.1 Using the Scale function (FC105)

Network 1: Scaling the value for an analog input

FC41 calls the Scale function (FC105) to adjust the value of the analog input proportionately between two values (the upper and lower limits).

Figure 1-1: FC41 (Network 1) – Scaling the analog input



Because there is no contact in the rung, the controller executes FC105 on every scan cycle:

1. FC105 reads the integer value for analog input stored in PIW320 (parameter IN). For this example, the state of I 1.0 (parameter BIPOLAR) determines whether the input value is bipolar or unipolar. Because the value of I 1.0 is 0, FC105 processes the analog input as a unipolar number (0 to 27648)
2. FC105 converts the integer value to a Real number.
3. FC105 scales the Real number to a value between 0.00 (parameter LO_LIM) and 400.00 (parameter HI_LIM):
   - If the function is executed without error, FC105 sets the ENO and Q1.1 to 1 (ON) and sets the RET_VAL (MW200) to W#16#0000 (hexadecimal).
   - If the input integer value is greater than 27648 or less than 0, FC105 sets the output (OUT) to 400.00 (HI_LIM) or 0.00 (LO_LIM), respectively. FC105 also sets the ENO and Q1.1 to 0 (OFF) and returns an error by setting the RET_VAL to W#16#0008 (hexadecimal).
4. FC105 stores the scaled value in MD220 (parameter OUT).

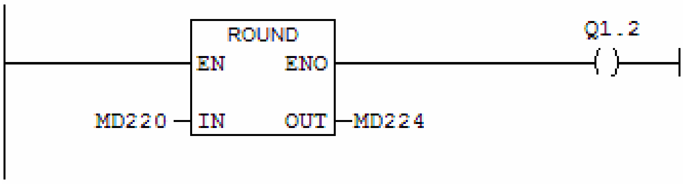
1.2 Ensuring that the scaled value is within a specified range

FC41 includes logic for testing that the scaled value falls within a range of operational values. For this example, FC41 turns on an output if the scaled value of the analog input is either above the upper range (300) or below a lower limit (100).

Note Because achieving an equal condition is more difficult when comparing two Real numbers than when comparing double integers, this example uses the Round instruction to convert the scaled value from a Real number to a double-integer value.

Network 2: Rounding a Real value to a Double Integer for comparison

Figure 1-2: FC41 (Network 2) – Converting a Real number to a Double Integer



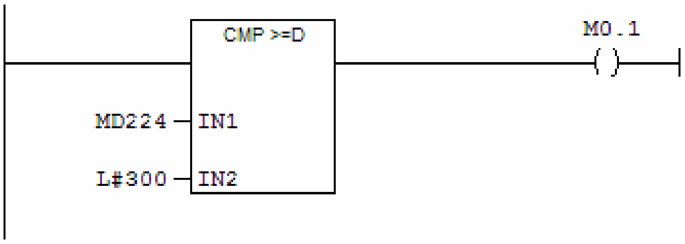
The Round instruction performs the following operations every scan:

1. Reads the content of MD220 (parameter IN) as a Real number
2. Converts the Real number to the closest double-integer value
3. Stores the result in MD224 (parameter OUT)

If an overflow occurs, the ROUND instruction sets ENO and the output (Q1.2) to 1 (ON).

Network 3: Checking for the upper limit of the operating range

Figure 1-3: FC41 (Network 3) – Checking for an out-of-range value (upper limit)

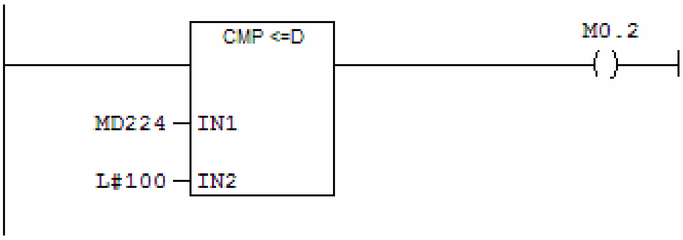


The Compare instruction (CMP <=D) checks to see whether the scaled value is above the higher limit of the operating range.

On every scan, the Compare instruction (CMP >=D) reads the double-integer value at MD224 (parameter IN1) and compares it to the constant value of 300 (parameter IN2). If IN1 is greater than or equal to IN2, the Compare instruction sets M0.1 to 1 (ON).

Network 4: Checking for the lower limit of the operating range

Figure 1-4: FC41 (Network 4) – Checking for an out-of-range value (lower limit)

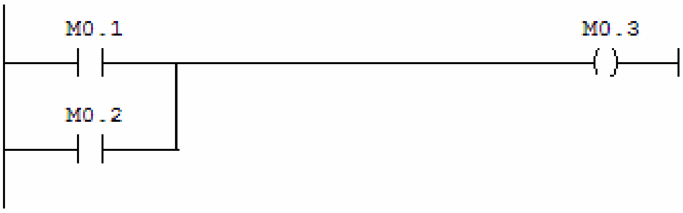


The Compare instruction (CMP <=D) checks whether the scaled value is below the lower limit of the operating range.

On every scan, the Compare instruction (CMP <=D) reads the double-integer value at MD224 (parameter IN1) and compares it to the constant value of 100 (parameter IN2). If IN1 is less than or equal to IN2, the Compare instruction sets M0.1 to 1 (ON).

Network 5: Reporting an out-of-range condition

Figure 1-5: FC41 (Network 5) – Reporting an out-of-range condition



By setting M0.3 to 1 (on), FC41 reports that an out-of range condition has been detected:

- M0.1 is 1 (ON) if network 3 identified a low out-of-limit value
- M0.2 is 1 (ON) if Network 4 identified a high out-of-limit value

If either M0.1 or M0.2 is 1 (ON), then M0.3 is set to 1 (ON).

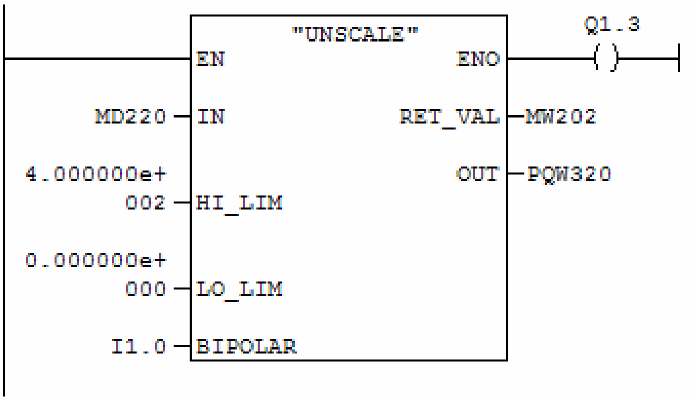
1.3 Using the Unscale function (FC106)

This example also uses the Unscale function (FC106) to convert a Real number value to an integer value.

Network 6: Unscaling Analog Output Values

FC41 calls the Unscale function (FC106) to convert the scaled Real number to an integer value proportionately between two values (the upper and lower limits).

Figure 1-6: FC41 - Network 6



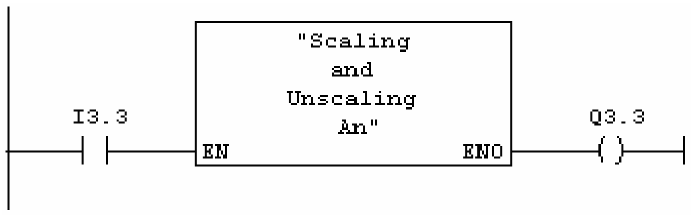
Because there is no contact in the rung, the controller executes FC106 on every scan cycle:

1. FC106 reads the Real number stored in MD220 (parameter IN). For this example, the state of I 1.0 (parameter BIPOLAR) determines whether the input value is bipolar or unipolar. Because the value of I 1.0 is 0, FC106 processes the Real number as a unipolar integer value (from 0 to 27648)
2. FC106 converts the Real number that is scaled between 0.00 (parameter LO_LIM) and 400.00 (parameter HI_LIM) to an integer value.
   - If the operation is executed without error, FC106 sets the ENO and Q1.3 to 1 (ON) and sets the RET_VAL (MW202) to W#16#0000 (hexadecimal)
   - If the input Real number is greater than 400.00 (HI_LIM) or less than 0.00 (LO_LIM), FC106 sets the output (OUT) to 27648 or 0, respectively. FC106 also sets the ENO and Q1.3 to 0 (OFF), and returns an error by setting the RET_VAL to W#16#0008 (hexadecimal)
3. FC106 stores the integer value in PQW320 (parameter OUT).

2 Using a Cyclic OB to call FC41 at a specific interval

By using a cyclic OB (OB30 to OB38), your program can perform a task or operation at a specific interval.

Figure 2-1: OB31 – Calling FC41



The sample program uses OB31 to call FC41 every 2 seconds (2000 ms):

1. After the controller goes to Run mode, OB31 starts every 2 seconds.
2. Turning on I 3.3 calls FC41 ("Scaling and Unscaling an Analog Input").
3. If FC41 is processed correctly:
   - Q3.3 = 1 (ON)
   - ENO = 1

3 Reference Information

3.1 Functional Descriptions for FC105 and FC106

Description of the Scale Function (FC105)

The SCALE function takes an integer value (IN) and converts it to a real value in engineering units scaled between a low and a high limit (LO_LIM and HI_LIM). The result is written in OUT. The SCALE function uses the equation:

OUT = [((FLOAT (IN) - K1)/(K2 - K1)) * (HI_LIM - LO_LIM)] + LO_LIM

The constants K1 and K2 are based upon whether the input value is BIPOLAR or UNIPOLAR.

- BIPOLAR: The input integer value is assumed to be between -27648 and +27648, therefore, K1 = -27648.0 and K2 = +27648.0
- UNIPOLAR: The input integer value is assumed to be between 0 and 27648, therefore, K1 = 0.0 and K2 = +27648.0

If the input integer value is greater than K2, the output (OUT) is clamped to HI_LIM, and an error is returned. If the input integer value is less than K1, the output is clamped to LO_LIM, and an error is returned.

Reverse scaling can be obtained by programming LO_LIM > HI_LIM. With reverse scaling, the value of the output decreases as the value of the input increases.

Description of the Unscale Function (FC106)

The UNSCALE function takes a real input value (IN) in engineering units scaled between a low and a high limit (LO_LIM and HI_LIM) and converts it to an integer value. The result is written to OUT. The UNSCALE function uses the following equation:

OUT = [((IN - LO_LIM) / (HI_LIM - LO_LIM)) * (K2 - K1)] + K1

The constants K1 and K2 are based upon whether the input value is BIPOLAR or UNIPOLAR.

- BIPOLAR: The output integer value is assumed to be between -27648 and +27648; therefore, K1 = -27648.0 and K2 = +27648.0
- UNIPOLAR: The output integer value is assumed to be between 0 and 27648; therefore, K1 = 0.0 and K2 = +27648.0

If the input value is outside the LO_LIM and HI_LIM range, the output (OUT) is clamped to the nearer of either the low limit or the high limit of the specified range for its type (BIPOLAR or UNIPOLAR) and an error is returned.

3.2 Modifying the start-intervals for a Cyclic OB (OB30 to OB38)

The S7 controller uses cyclic interrupt OBs to interrupt the main program scan (OB1) at specified intervals. (The time at which the interval starts is the mode transition from STOP to RUN.) Each of the cyclic interrupt OBs has a default execution interval, as shown in the following table. The execution interval value for each cyclic interrupt OB is configurable.

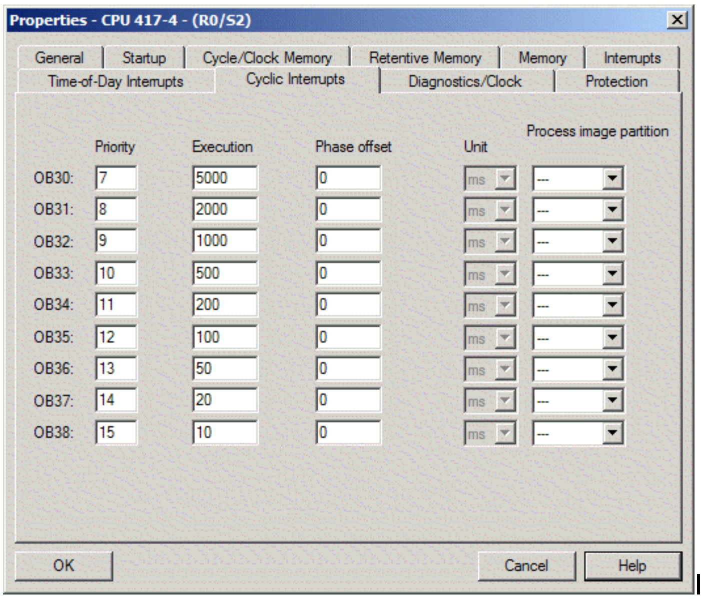
Table 3-1: Cyclic Interrupt OBs

Cyclic Interrupt OB	Interval in ms	Priority Class
OB30	5000	7
OB31	2000	8
OB32	1000	9
OB33	500	10
OB34	200	11
OB35	100	12
OB36	50	13
OB37	20	14
OB38	10	15

When you use STEP 7 to configure the hardware for the project, you can modify the execution rate of cyclic interrupt OBs:

1. Double-click the CPU module to display the CPU Object Properties.
2. Select the Cyclic Interrupts tab.
3. Enter the new execution interval (in milliseconds) and click OK.

Figure 3-1: Cyclic Interrupts Tab of the Object Properties for the CPU Module



	Priority	Execution	Phase offset	Unit	Process image partition
OB30:	7	5000	0	ms	---
OB31:	8	2000	0	ms	---
OB32:	9	1000	0	ms	---
OB33:	10	500	0	ms	---
OB34:	11	200	0	ms	---
OB35:	12	100	0	ms	---
OB36:	13	50	0	ms	---
OB37:	14	20	0	ms	---
OB38:	15	10	0	ms	---