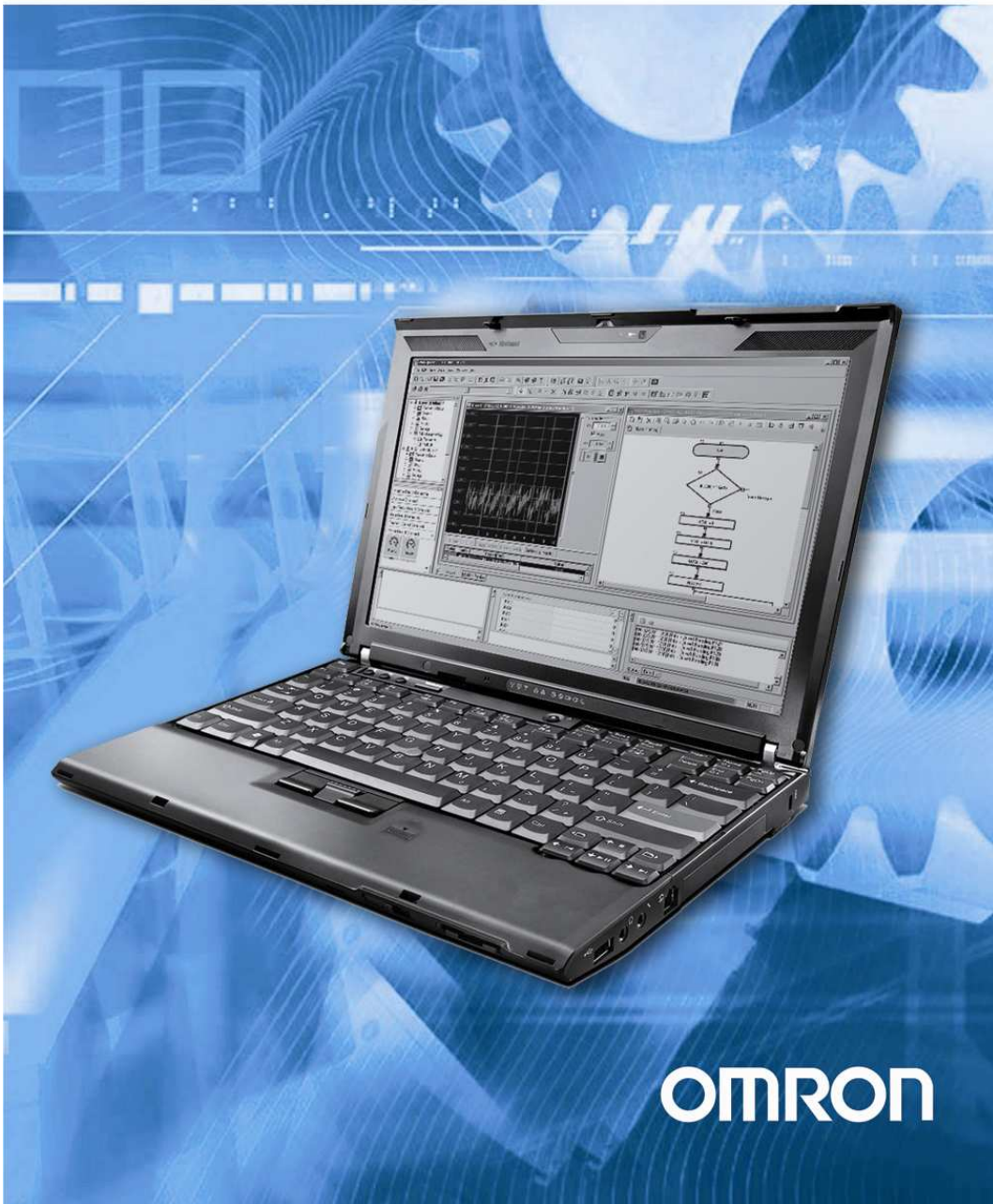


CX-Drive Drive Programming

USER'S MANUAL



OMRON

Table of Contents

| | |
|---|-----------|
| 1- Introduction | 5 |
| 1-1 Handling of this Instruction Manual..... | 5 |
| 1-2 Safety Instruction | 5 |
| 1-3 Preparation and System configuration..... | 6 |
| 2- Specifications | 7 |
| 3- Drive Programming Editor..... | 8 |
| 3.1- Saving and loading programs | 9 |
| 3.2- Editor..... | 9 |
| 3.3- Toolbar | 10 |
| 3.4- Shortcut Keys..... | 11 |
| 3.5- Designer Area | 11 |
| 3.6- Toolbox window | 13 |
| 3.7- Block Parameters window..... | 14 |
| 3.8- Properties window..... | 15 |
| 3.9- Output window | 16 |
| 3.10- Creating a program | 16 |
| 4- Drive Program structure..... | 17 |
| 4.1- Tasks..... | 17 |
| 4.2- Subroutines | 18 |
| 5- Drive Programming user variables..... | 19 |
| 5.1- Initial Data | 19 |
| U(00) to U(31) or User parameters..... | 19 |
| UL(00) to UL(07) or Internal User parameters..... | 19 |
| 5.2- Setting Variables | 19 |
| Set-Freq | 19 |
| ACCEL | 19 |
| DECEL | 20 |
| 5.3- Inverter Monitor Variables | |
| FM..... | 20 |
| Iout | 20 |
| Dir..... | 20 |
| PIB-FB | 20 |
| F-CNV | 20 |
| Tmon..... | 21 |
| Vout..... | 21 |
| Power | 21 |
| Run-Time | 21 |
| On-Time | 21 |
| UMon(0) to UMon(2)..... | 21 |
| POS | 22 |
| ERR-CNT..... | 22 |
| ERR(1)-ERR(6)..... | 22 |
| DCV | 22 |
| STATUS..... | 22 |
| 5.4- Terminal Variables | 23 |
| X(00)-X(09) | 23 |
| Xw | 23 |
| Y(00)-Y(05) | 23 |
| Yw | 23 |
| XA(0)-XA(2) | 23 |
| YA(0)-YA(2) | 23 |
| UB(00)-UB(07) | 24 |
| UBw | 24 |
| TC(0)-TC(7) | 24 |
| TD(0)-TD(7) | 24 |
| TDw..... | 24 |

| | |
|---|-----------|
| 5.5- Digital input Functions..... | 25 |
| 5.6- Digital Output Functions..... | 26 |
| 6- Drive Programming Instructions | |
| 6.1- Control Commands | 28 |
| Entry..... | 28 |
| End..... | 28 |
| Call..... | 28 |
| Sub..... | 28 |
| End Sub | 29 |
| Go To | 30 |
| On Trip | 31 |
| If..... | 32 |
| Ifs/ Else / End If..... | 33 |
| Select / Case / End Select | 34 |
| For / Next | 35 |
| While / Wend..... | 36 |
| Until / Loop..... | 37 |
| Wait..... | 38 |
| 6.2- Arithmetic and Logic Commands | 39 |
| = (Substitution)..... | 39 |
| Addition | 39 |
| Subtraction | 40 |
| Multiplication | 40 |
| Division | 41 |
| Mod | 41 |
| Abs..... | 42 |
| And..... | 43 |
| Or..... | 44 |
| XOr..... | 45 |
| Not | 46 |
| Inc | 47 |
| Dec..... | 48 |
| 6.3- Input/Output Control Commands | 49 |
| Var = X(i)..... | 49 |
| Var = Xw | 50 |
| Y(i) = value | 51 |
| Yw = value | 52 |
| func = value | 53 |
| Var = func..... | 54 |
| Var = UB(i) | 55 |
| Var = UBw..... | 56 |
| UB(i) = value | 57 |
| UBw = value..... | 58 |
| 6.4- Timer Control Commands..... | 59 |
| Delay | 59 |
| Timer Set | 60 |
| Timer Off | 61 |
| 6.5- Parameter Control Commands | 62 |
| ChgParam..... | 62 |
| MonParam | 63 |
| EepWrt | 64 |
| RtcSet | 65 |
| 6.6- Inverter Control Commands..... | 66 |
| Run-FW..... | 66 |
| Run-RV | 66 |
| Stop..... | 66 |
| Set-Freq | 66 |
| Trip | 68 |
| Accel | 69 |
| Decel..... | 69 |
| 7- Drive Programming specific trips and Troubleshooting..... | 70 |

| | |
|--|-----------|
| 8- Drive Programming Parameters – General Precautions | 71 |
| 8-1 Parameters list affected by setting order | 71 |
| 8-2 Parameters list affected by Rated Current (%)..... | 71 |
| 8-3 Parameters list affected by PID enabled/disabled | 72 |

1-Introduction

This Instruction Manual explains how to use the *Drive Programming* software for the Omron MX2/RX Series Inverter. Be sure to read this Instruction Manual carefully before using *Drive Programming*, and keep it on hand for future reference.

1-1 Handling of this Instruction Manual

- The contents of this Instruction Manual are subject to change without prior notice.
- No part of this Instruction Manual may be reproduced in any form without the publisher's permission.
- If you find any incorrect description, missing description or have a question concerning the contents of this Instruction Manual, please contact the publisher.

1-2 Safety Instruction

Be sure to read this Instruction Manual, Inverter Instruction Manual, and appended documents thoroughly before using *Drive Programming* and the inverter.

Before creating user programs for the inverter, also refer to the Inverter Instruction Manual and configuration software (CX-Drive) Instruction Manual for the necessary related Knowledge, and ensure you understand and follow all safety information, precautions, and operating and handling instructions for the correct use of the inverter.

Always use the inverter strictly within the range of specifications described in the Inverter Instruction Manual and correctly implement maintenance and inspection to prevent fault from occurring.

When using the inverter together with optional products, also read the manuals for those products. Note that this Instruction Manual and the manual for each optional product to be used should be delivered to the end user of the inverter.

In this instruction manual you can find WARNINGS along the instructions

WARNING: Indicates that incorrect handling may cause hazardous situation, which may result in serious personal injury or death.

1-3 Preparation and System configuration

To create user programs with Drive Programming function of the inverter, you must prepare the following devices and software:

- (1) MX2, RX inverter
- (2) Personal computer (PC) (Windows System)
 - 32-bit PC: Windows XP SP3, Windows Vista (any service pack) and Windows 7.
 - 64-bit PC: Windows Vista (any service pack) and Windows 7.
- (3) Optional programming software CX-Drive
 - MX2 inverter: CX-Drive version 2.0x or higher.
 - RX inverter: CX-Drive 2.3x or higher.
- (4) Optional PC-inverter connection cable. For MX2 it is a USB cable, For RX, the converter cable USB to RJ-45 is required. Item codes:
 - Item code name for MX2: **AX-CUSBM002-E**
 - Item code name for RX (2 option cables):
 - **3G3AX-PCACN2**, or
 - **USB CONVERTERCABLE**




RX

Inverter port: Operator-connection port RJ-45.

MX2

Inverter port: USB connector

The following figure shows the basic system configuration for programming.

| Optional programming software CX-Drive | Windows personal computer | Optional PC-Inverter cable | MX2 or RX Inverter |
|--|---|--|---|
|  MX2: CX-Drive 2.0x or higher RX: CX-Drive 2.3x or higher |  | - For MX2: · AX-CUSBM002-E - For RX (2 options): · 3G3AX-PCACN2 , or · USB-CONVERTERCABLE |  |

Install CX-Drive on your Windows personal computer, and connect the personal computer to the inverter (MX2 or RX) via the PC-inverter connection cable.

After completing these preparations, you can operate Drive Programming Editor to create a user program and download it to the inverter.

The table below lists the main functions of Drive Programming Editor.

| Function | Description |
|---------------------------|--|
| Programming Editor | Supports the input, editing, saving, reading, and printing of user programs |
| Compilation | Compile and edit a user program |
| Downloading and uploading | Downloads a user program to the inverter Uploads a user program from the inverter |


2- Specifications

The table below lists the programming-related specifications of the Drive Programming function.

| Item | Specification | | |
|---|--|---|---|
| Language specification | Programming language | Flow Chart language | |
| | Input device | Windows personal computer (OS: Windows XP-SP3, Windows Vista, Windows 7) | |
| | Max. program size | 1024steps (The internal storage capacity of the inverter is 1024 steps or 6 Kilobytes.) | |
| | Programming support function (programming software) | -Editing (on Windows) / - Display (on Windows) -Program syntax check (on Windows) -Downloading, uploading, and full clearance of program | |
| | Execution format | Execution by interpreter in an execution cycle of 2ms per instruction (possible subroutine call with nesting in up to 8 layers) | |
| Input/output-related functions | External input | Contact Signal | 24v open – collector input (using intelligent input terminals) RX: Assign to the PRG terminal / Always run |
| | | Program run signal input | MX2: Assign to the PRG terminal / Always run |
| | | Multifunction terminals | RX: Up to 8 terminals (X(00) to X(07)) |
| | | | MX2: Up to 8 terminals (X(00) to X(07)) |
| | General-purpose analog input | XA(0): 0 to 10V (O terminal) | |
| | | XA(1): 4 to 20mA (O terminal) | |
| | | XA(2): 0 to 10V (O2 terminal) (Only RX) | |
| | External Output | General-purpose output terminal | RX: Up to 6 terminals (Y(00) to Y(05)) |
| | | | MX2: Up to 3 terminals (Y(00) to Y(02)) |
| | | General-purpose analog output | YA (0): Assignable to the EO terminal (FM terminal for RX) |
| YA (1): Assignable to the AM terminal YA (2): Assignable to the AMI terminal (Only RX) | | | |
| Instructions | (1) Program control instructions -Loop (For) / - Unconditional branching (Goto) / -Time control (Wait) -Conditional branching (If Then, Ifs Then Else, Select Case, Until, and While) -Subroutine (call, sub) / - Others (Entry, End, Sub, End Sub, Inc, and Dec) (2) Arithmetic instructions -Arithmetic operation (+, -, *, /) / - Remainder (Mod) / -Substitution (=) -Absolute value (Abs) / - Logic operation (Or, And, Xor, and Not) (3) Input/Output control -General-purpose input/output (bit input, word input, bit output, and word output) - Reading of inverter input terminal. (4) Timer control: - Delay operation / -Timer control (5) Parameter control: - Rewriting of parameters by reselecting code on the operator's display | | |
| | Reserved Words | User-defined variable | U (00) to U (31) (32 variables) |
| Internal user variable | | UL (00) to UL (07) (8 variables) | |
| Set frequency | | SET-Freq | |
| Acceleration time | | ACCEL | |
| Deceleration time | | DECEL | |
| Monitoring variable | | FM, Iout, Dir, PID-FB, F-CNV, Tmon, Vout, Power, RUN-Time, ON-Time, PlsCnt (Only RX), POS, STATUS, DCV, ERR CNT, ERR(1), ERR(2), ERR(3), ERR(4), ERR(5), and ERR(6) | |
| Bit commands | | FW, RV, CF1, CF2, CF3, CF4, JG, DB, SET, TCH, FRS, EXT, USP, CS, SFT, AT, RS, STA, STP, F/R, PID, PIDC, UP, DWN, UDC, OPE, SF1, SF2, SF3, SF4, SF5, SF6, SF7, OLR, TL, TRQ1-2, BOK, LAC, PCLR, ADD, F-TM, ATR, KHC, AHD, CP1-3, ORL, ORG, SPD, RS485, HLD, ROK, DISP. | |
| Output Functions | | RUN, FA1, FA2, OL, OD, AL, FA3, OTQ, UV, TRQ, RNT, ONT, THM, BRK, BER, ZS, DSE, POK, FA4, FA5, OL2, ODc, OIdc, FBV, NDc, LOG1, LOG2, LOG3, WAC, WAF, FR, OHF, LOC, IRDY, FWR, RVR, MJA, WCO, WCOI, FREF, REF, SETM, EDM. | |
| General-purpose input contact | | RX : X (00) to X (07) (8 contacts) | |
| | | MX2 : X(00) to X(07) (8 contacts) | |
| Extended IO option input contact | | RX : ----- | |
| | | MX2 : X(08) to X(09) (2 contacts) | |
| General-purpose output contact | | RX : Y(00) to Y(05) (6 contacts) | |
| | | MX2 : Y(00) to Y(02) (3 contacts) | |
| Extended output contact | | -- | |
| | | MX2 : Y(03) to Y(05) (3 contacts) | |
| Internal user contact | | UB (00) to UB (7) (8 contacts) | |
| Internal timer contact | TD (0) to TD (7) (8 counter contacts) | | |
| Internal timer counter | TC (0) to TC (7) (8 counters) | | |
| Inverter input/output | Specification by code on the remote operator's display | | |
| User Monitor | UMon(00) to UMon(02) (3 user monitors) | | |
| User trip | Makes the inverter trip (10 trips) | | |

3- Drive Programming Editor

Drive Programming Editor allows the user to design drive programs in an intuitive way. CX-Drive provides a way to create drive programs, compile them, transfer them to and from the drive, start and stop their execution, and other related tasks.

You can open this function by clicking on Drive Programming in the workspace of a drive which supports it, or selecting *Program | Program Editor* from the Drive menu, or with the CX-Drive toolbar button .

Please create a new CX-Drive File by clicking on the menu *File | New*. The New Drive window will appear (Image 1). Select the Drive Type and press OK button. Then it will appear on the Workspace (Image 2).

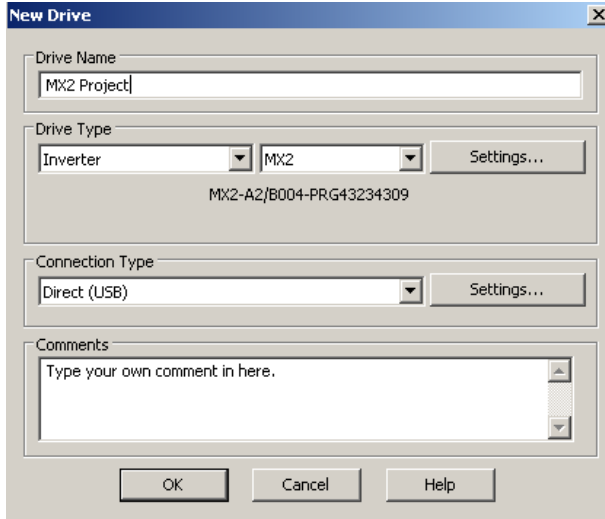


Image 1- New Drive window

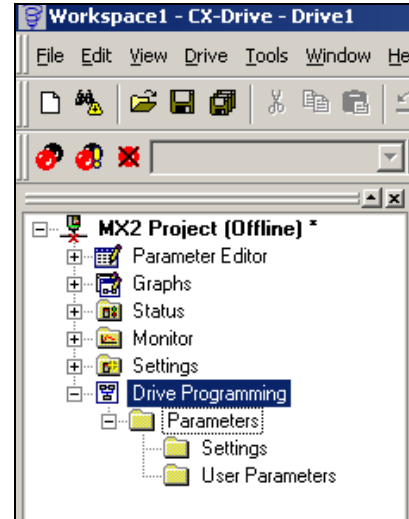
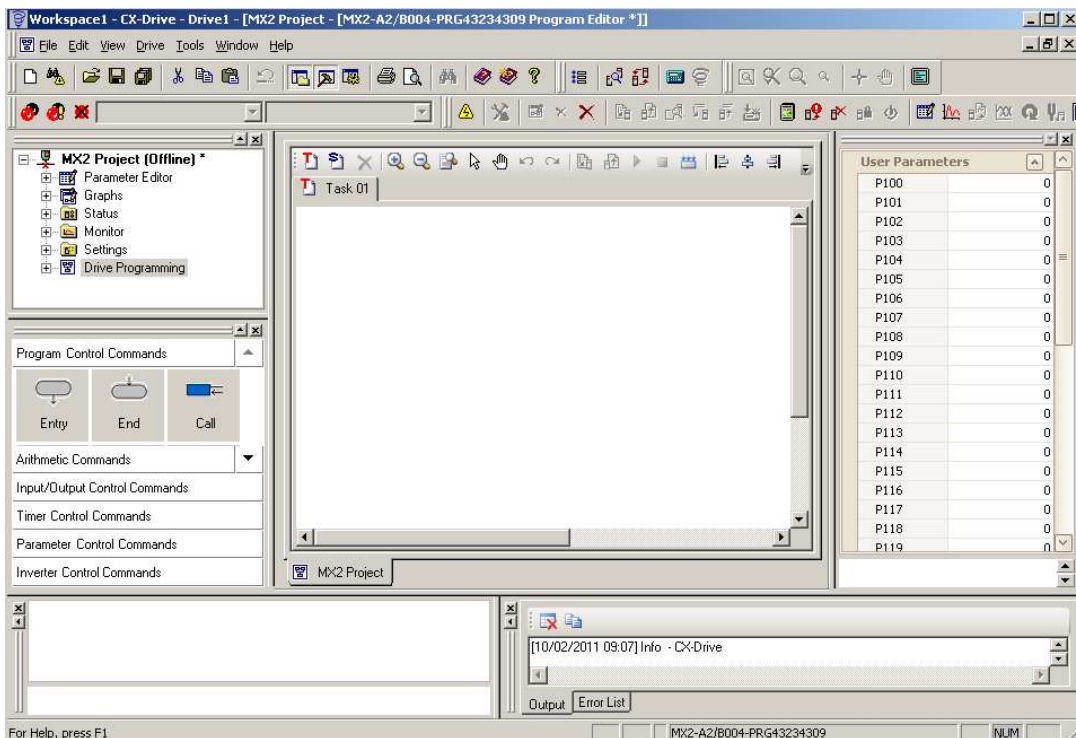


Image 2- CX-Drive Workspace

Making double-click to the Drive Programming option, the Drive Programming Editor will appear.



3-1 Saving and loading programs

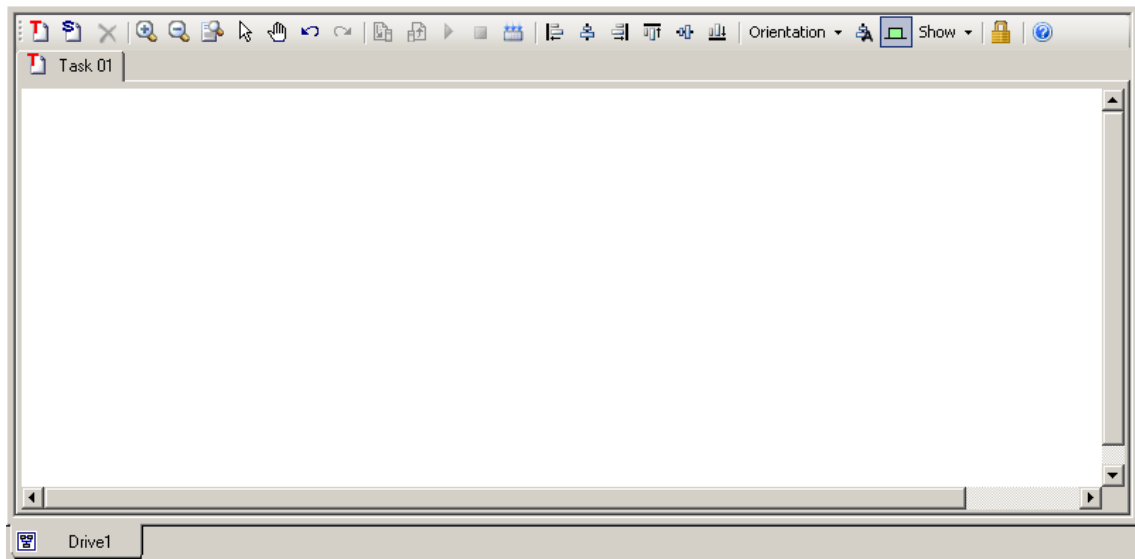
A drive program is automatically saved when the CX-Drive drive document which contains it is saved. When a CX-Drive document is opened, the drive program which it contains, if any, is automatically loaded. You can display it by opening the Program Editor.

Alternatively, you can export a drive program, to save it independently of other information of the drive. To do so, use the *Program | Export Program* command in the Drive menu. Enter the name of the file to be used. The file will be saved with extension *driveprogram*.

A drive program can be imported with the *Program | Import Program* command in the Drive menu.

3-2 Editor

The Program Editor is the main window of the Drive Programming function.



The window area consists of a toolbar with common commands, and a designer area where the program is displayed as a flowchart.

3-3 Toolbar

The Program Editor window contains the following commands:

| Commands | Image | Description |
|--------------------------------|-------|---|
| New task | | It allows creating a new task for the program, up to the maximum number of tasks allowed. Tasks are parts of the program which are executed independently of each other. |
| New Subroutine | | It allows creating a new subroutine. A subroutine is a part of the program which is called from a task. |
| Delete Current Tab | | It deletes the current Task or Subroutine. |
| Zoom in | | It increases the zoom level. |
| Zoom out | | It decreases the zoom level. |
| Zoom Reset | | It restores the zoom to its initial value. |
| Select Mode | | It allows the user to select one or more elements of the program, by click-and-drag with the mouse cursor. This mode is active by default. |
| Pan Mode | | It allows the user to move the extent of the view. in any direction while keeping the same scale, by click-and-drag. |
| Undo | | It reverts the latest change. |
| Redo | | It recovers the most recently undone change. |
| Transfer to Drive | | It compiles the program and, if there are no errors, transfer it to the drive. |
| Transfer from Drive | | It transfers the program from the drive to the Program Editor. |
| Start | | It starts the program in the drive. CX-Drive will first compare it with the program currently being edited, to make sure that they are the same. If they differ, the program will not be started. |
| Stop | | It stops the program in the drive. This action is done regardless of whether the program in the drive is the same that in the program designer. |
| Compile | | It compiles the program which is currently being designed. Compile errors and warnings will be reported as tool tips in the blocks in the flowchart. |
| Horizontal Align Left | | It aligns horizontally the left sides of the selected blocks. |
| Horizontal Align Middle | | It aligns horizontally the middles of the selected blocks. |
| Horizontal Align Right | | It aligns horizontally the right sides of the currently selected blocks. |
| Vertical Align Top | | It aligns vertically the top sides of the selected blocks. |
| Vertical Align Middle | | It aligns vertically the middles of the selected blocks. |
| Vertical Align Bottom | | It aligns the bottom sides of the selected blocks. |
| Orientation | | It selects a preferred orientation for connecting the blocks. |
| Auto-arrange | | It arranges the elements of the flowchart automatically in the currently selected orientation. |
| Show contacts | | It toggles display/hide of the contacts of the blocks, which are placeholders for the beginning and ending of arrow connections |
| Show | | It allows you to select a display style of the program. (Text only, Icon Only, Icon and Text, or Name, Icon and Arguments). |
| Password | | It allows you to set, change or remove the program password. |
| Help | | It displays the Drive Programming help. |

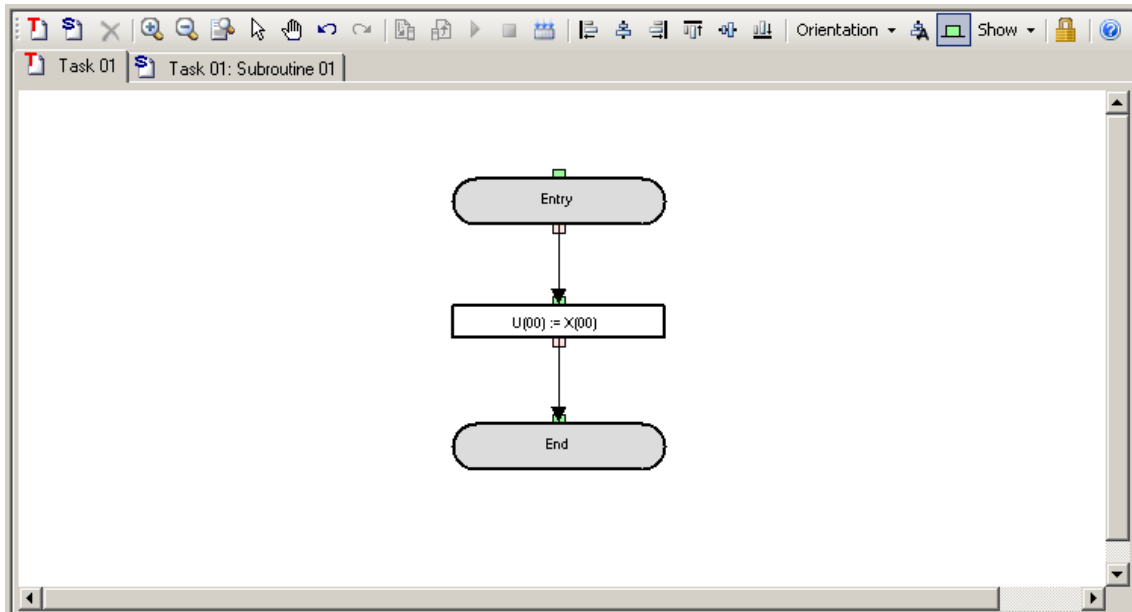
3-4 Shortcut Keys

The following Keyboard shortcuts can be applied to the designer area.

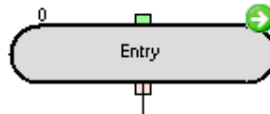
- Ctrl + X: Cut
- Ctrl + C: Copy
- Ctrl + V: Paste
- Ctrl + Z: Undo
- Ctrl + Y: Redo
- Ctrl + A: Select All
- Ctrl + L: Lock
- Ctrl + P: Pin
- Tab: Select Next
- Shift + Tab: Select Previous
- Arrow Keys: Move selected element
- Home, End, Page Up, Page Down: Navigate through the graph
- +: Zoom In
- -: Zoom Out

3-5 Designer Area

The designer area will display the current design of the program.



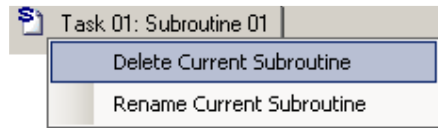
This area may have different pages, organized in tabs. Each tab is either a Task or a Subroutine. The designer is created with one default tab, which is a Task. When a program is compiled without error, an icon with a circled green arrow highlights the starting point of each task.



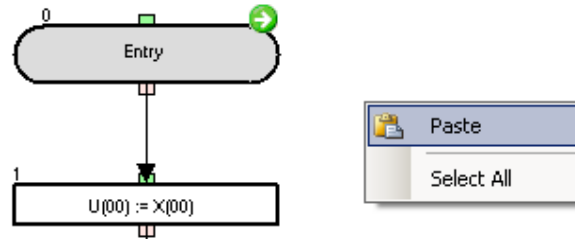
For programs compiled with errors, a red icon with an exclamation mark identifies the erroneous blocks. Placing the mouse on the error icon displays the compile error, which can also be seen in the Error List.



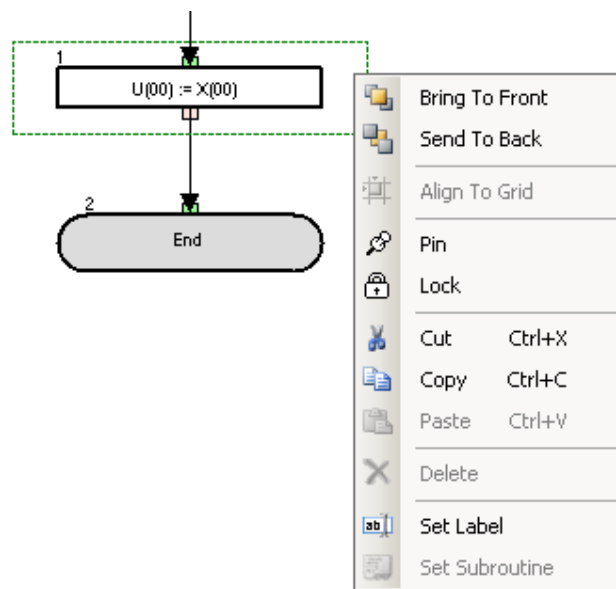
A Task or Subroutine may be deleted, or renamed, by right-clicking on the tab title.



Right-clicking on an area which is not an element of the flowchart displays a popup menu which allows you to Paste elements that you have previously copied, or to select all the elements.



Right-clicking on an element selects it and shows a popup menu with more options



The available menu commands are described below.

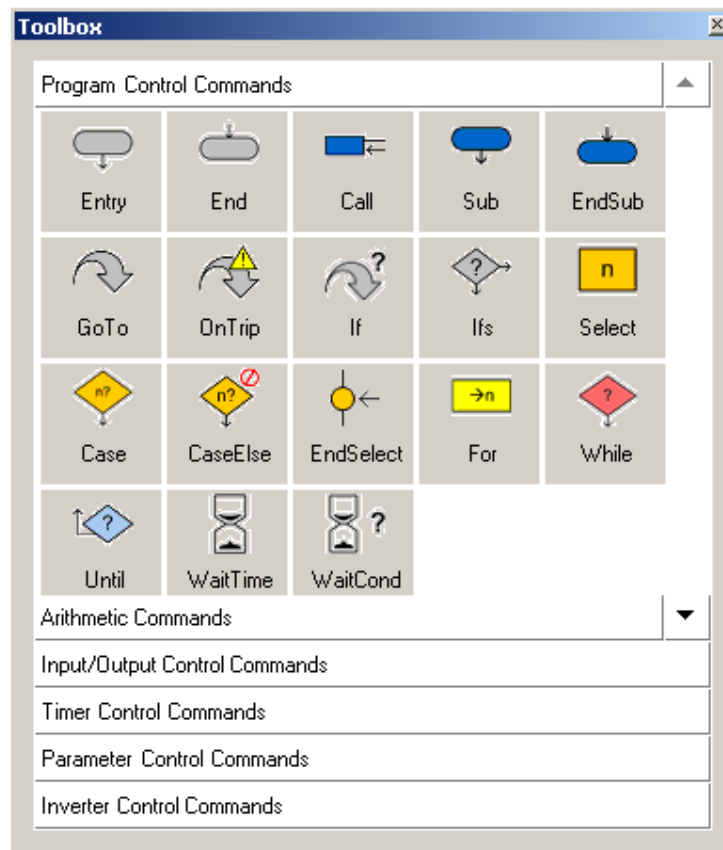
- *Bring To Front* places the element graphically in front of other elements.
- *Send To Back* places the element graphically in back of other elements.
- Pin fixes the element to its current position in the graph. It will not be moved in click-and-drag operations.
- Lock acts like Pin and, besides, sets the properties of the element as read-only.
- Cut deletes the element and saves it in the clipboard, for further pasting.
- Copy saves the element in the clipboard, for further pasting.
- Paste put the contents previously copied in the clipboard into the design area. Note that after copying elements, you can also paste them to other contexts; for example, as images in a Microsoft Office application.

3-6 Toolbox window

The Toolbox window allows you to add blocks to the Program Designer by drag and drop. It displays the blocks supported for a particular drive, organized in categories.

The Toolbox is displayed when Drive Programming is entered. You can also show or hide it by clicking on *Drive Programming* | Toolbox in the View menu.

The Toolbox is displayed by default docked at the rightmost side of CX-Drive. You can resize it as needed to better display its elements. Also, you can toggle its docking by right clicking near the window's edges.



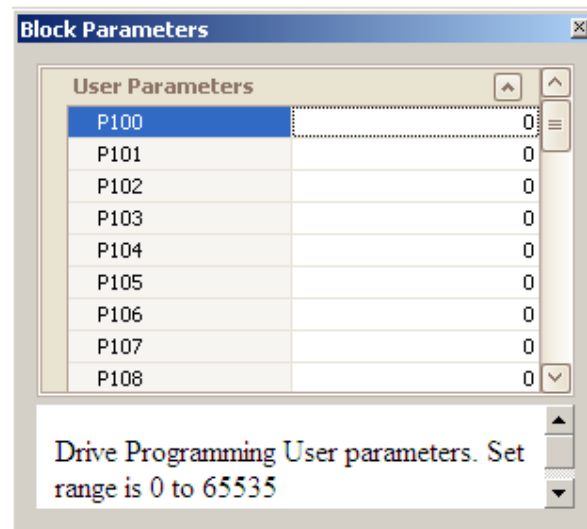
You can also choose its displays style by right-clicking on it with the mouse. Three styles are available: Large Icons, Small Icons, and List. In any style, placing the mouse cursor on a block will show a short help text for it.

Click on any category title to display the blocks which belong to that category.

3-7 Block Parameters window

The Block Parameters window allows the user to edit drive program parameters which act as variables of the program. The parameters are organized in categories. Block parameters is displayed when Drive Programming is entered. You can also show or hide it by clicking on Drive Programming | Block Parameters in the View menu.

Block Parameters is displayed by default docked at the rightmost side of CX-Drive. You can resize it as needed to better display its elements. Also, you can toggle its docking by right clicking near the window's edges.



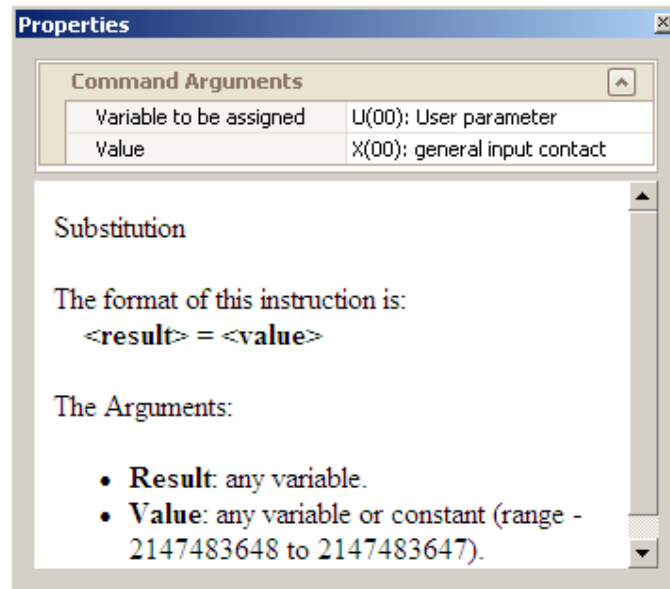
To change the value of a block parameter, place the cursor at its row and click on the edition box to the right of its name. Enter the new value. CX-Drive will warn you if the value exceeds the valid range. At the lower part of the window, a help text for the block parameters is displayed.

3-8 Properties window

The Properties window allows the user to edit the properties of the drive program block which is currently selected in the Program Editor.

Properties are displayed when Drive Programming is entered. You can also show or hide it by clicking on Drive Programming | Properties in the View menu.

Properties are displayed by default docked at the rightmost side of CX-Drive. You can resize it as needed to better display its elements. Also, you can toggle its docking by right clicking near the window's edges.



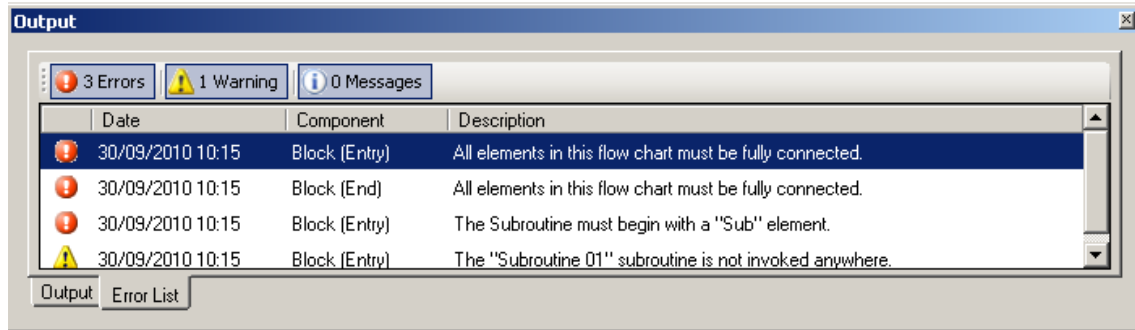
To change one block command argument, place the cursor at its row and click on the edition box to the right of its name.

- If the block argument has options, a second click of the mouse will unfold the available options for you to select.
- If the block argument does not have options, clicking on its current value will enable you to change it by typing a new one. CX-Drive will warn you if the value exceeds the valid range.

If the block argument can have both an option and a custom value, clicking on the unfold sign at the right of the cell will unfold the available options, whereas clicking anywhere in the cell text, you will be able to edit it.

3-9 Output window

It shows the compilation errors and warnings of the currently edited drive program after it is compiled. Errors will prevent the program to be correctly compiled. Warnings will allow compilation, but advise customer of abnormal conditions.



- The Error(s) button toggles displaying error in the list.
- The Warning(s) button toggles displaying warnings in the list.
- The Message(s) button toggles displaying informative message in the list.

Messages in the list show the following information:

- Date: The date and time when the error was generated.
- Component: Identifies the element with an error.
- Description: The text of the error or warning message.

The list is automatically cleared every time a Compile is done.

3-10 Creating a program

Follow the steps described below to create a drive program.


1. Open the Program Editor. The Drive Programming auxiliary windows (Toolbox, Block Parameters, Properties and Error List) will be displayed automatically.
2. Drag each block of the program from the Toolbox window to the Program Editor.
3. After dragging a block, edit its properties by clicking on it and edit the arguments in the Properties window.
4. Connect the blocks accordingly.
5. Edit the drive program variables in the Block Parameters window.
6. You may now compile the program, transfer it to the drive, export it, etc.

Alternatively, you can connect to a drive which has a program and transfer it, following the simple steps described below.

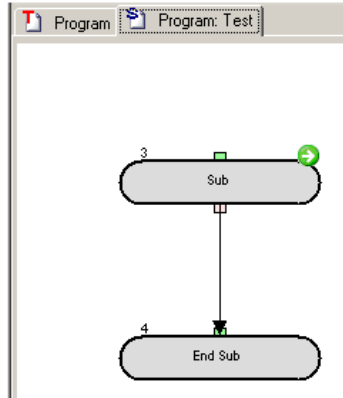
1. Open the Program Editor. The auxiliary Drive Programming windows (Toolbox, Block Parameters and Properties) will be displayed automatically.
2. Click the Transfer from Drive button in the program Editor Toolbar. The program will be transferred from the drive and automatically displayed in the Program Editor designer area.
3. You may now edit the program, compile it, transfer it to the drive, export it, etc.

When a drive program is present, you can also transfer it from and to the drive with the Transfer to Drive and Transfer from Drive buttons of the CX-Drive toolbar. In this case, a message dialog will ask you whether to transfer the parameters, the program or both.

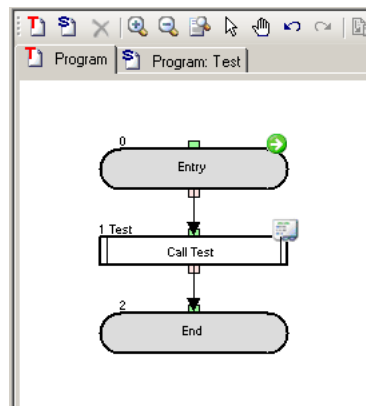
4-2 Subroutines

Subroutines are useful to organize your program into parts of code that you can reuse in other programs or in the same program. For insert a subroutine press the button  and a new subroutine will appear. Similarly to Tasks, you can delete or rename a subroutine.

Every subroutine must begin with the **Sub** block, and end with the **EndSub** Control Command.



The subroutine is executed via the call command with the subroutine name.



It is only possible to call a subroutine that is associated with the task. to be used with other task, a copy of the subroutine is necessary on the task.

5- Drive Programming user variables

5-1 Initial Data

U(00) to U(31) or User parameters

| U(00) to U(31) | Description | Range of values | Default | Unit | Data size | Attribute |
|----------------|---------------|-----------------|-----------------------------|------|-----------------|-----------|
| | User variable | 0 to 65535 | Data stored in P100 to P131 | - | Unsigned 1-word | R/W |

User variables are the general-purpose functions that can be used as unsigned 1-word. The data written from a drive program to the user-defined variables is not stored in the inverter's EEPROM. The variables will restore the initial settings when the inverter power is turned off. The user-defined variables correspond to inverter parameters "P100" to "P131". You can also change the settings of user-defined variables from the digital operator. The changes made from the digital operator will be stored in EEPROM. This is also possible to emulate from drive programming by using the EepWrt command.

The variables P129 to P131 (U(29) to U(31)) are saved at power down of the inverter automatically. This function may not work under heavy load (motor output current) or too small inverter (low capacity in DC bus). In case of trouble it is recommended to disable the inverter output to preserve the energy in the capacitors.

UL(00) to UL(07) or Internal User parameters.

| UL(00) – UL(07) | Description | Range of values | Default | Unit | Data size | Attribute |
|-----------------|------------------------|-------------------------|---------|------|---------------|-----------|
| | Internal user variable | -2^{31} to $2^{31}-1$ | 0 | - | Signed 2-word | R/W |

Internal user variables are the general-purpose functions that can be used as unsigned 2-word variables, for example, to temporarily store arithmetic operation results. The initial values can be set via the initial program data.

5-2 Setting Variables

| Set-Freq | Description | Range of values | Default | Unit | Data size | Attribute |
|----------|--------------------------|-----------------|---------|---------|-----------------|-----------|
| | Output frequency setting | 0 to 40000 | 0 | 0.01 Hz | Unsigned 1-word | R/W |

When A001=7 (Freq. ref. from Drive Programming), it is the frequency set point for the inverter. Always it reflects the reading of parameter F001, regardless the setting of A001. This variable is not stored in the inverter EEPROM. It will be restored to initial setting after power cycle. When the inverter receives an operation command (FW=1 or RV=1), it accelerates the motor up to the frequency that was set last.

| ACCEL | Description | Range of values | Default | Unit | Data size | Attribute |
|-------|---------------------------|-----------------|---------------|----------|-----------------|-----------|
| | Acceleration time setting | 1 to 360000 | <u>Note 1</u> | 0.01 sec | Unsigned 2-word | R/W |

This variable can be used to read and write the motor acceleration time in the inverter. It is enabled only when the setting of accel/decel time input selection (P031) is "03" (PRG). (Please note that it does not correspond to the setting of inverter parameter "F002"). The data written to this variable is not stored in the inverter's EEPROM. It restores initial value after power cycle.

Note 1: By default (when the inverter power is turned on), the acceleration time follows the setting of the inverter parameter "F002", "F202", or "F302". For details, refer to the Inverter Instruction Manual.

Note 2: When a program writes a value to this variable, the value is reflected in the inverter in a 40-ms cycle, which conforms to the standard inverter specifications.

| DECCEL | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|---------------------------|-----------------|---------------|----------|-----------------|-----------|
| | Deceleration time setting | 1 to 360000 | <u>Note 1</u> | 0.01 sec | Unsigned 2-word | R/W |

This variable can be used to read and write the motor deceleration time in the inverter. The deceleration time setting using this variable is enabled only when the setting of accel/decel time input selection (P031) is "03" (PRG). (The setting of this variable does not correspond to the setting of inverter parameter "F003"). The data written to this variable is not stored in the inverter's EEPROM. This variable will restore the initial setting when the inverter power is turned off.

Note 1: By default (when the inverter power is turned on), the deceleration time follows the deceleration (1) time setting "F003", "F203" or "F303". For details, refer to the Inverter Instruction Manual.

Note 2: When a program writes a value to this variable, the value is reflected in the inverter in a 40-ms cycle, which conforms to the standard inverter specifications.

5-3 Inverter Monitor Variables (This units does not always corresponds with the display units)

| FM | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|--------------------------|-----------------|---------|---------|-----------------|-----------|
| (d001) | Output frequency monitor | 0 to 40000 | - | 0.01 Hz | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the output frequency monitor (d001). This variable is read-only.

| lout | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|------------------------|-----------------|---------|--------|-----------------|-----------|
| (d002) | Output current monitor | 0 to 9999 | - | 0.01 % | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the output current monitor (d002). The monitored data indicates the ratio of present output current to rated current of the inverter. This variable is read-only. For details, refer to the Inverter Instruction Manual.

| Dir | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|----------------------------|--|---------|------|-----------------|-----------|
| (d003) | Rotation direction monitor | 0: Stop 1: Normal rotation 2: Reverse rotation | - | - | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the rotation direction monitor (d003). This variable is read-only.

| PID-FB | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|--|-----------------|---------|--------|-----------------|-----------|
| (d004) | Process variable (PV), PID feedback monitoring | 0 to 9990000 | 0 | 0.01 % | Unsigned 2-word | R |

The data monitored with this variable corresponds to the data monitored by the process variable (PV), PID feedback monitor (d004). This variable is read-only.

| F-CNV | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|---------------------------------|-----------------|---------|------|-----------------|-----------|
| (d007) | Scaled output frequency monitor | 0 to 3996000 | - | 0.01 | Unsigned 2-word | R |

The data monitored with this variable corresponds to the data monitored by the scaled output frequency monitor (d007). This variable is read-only.

| Tmon | Description | Range of values | Default | Unit | Data size | Attribute |
|-------------|----------------|-----------------|---------|------|--------------------|-----------|
| (d012) | Torque monitor | -300 to 300 | - | % | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the torque monitor (d012). This variable is read-only.

| Vout | Description | Range of values | Default | Unit | Data size | Attribute |
|-------------|------------------------|-----------------|---------|------|--------------------|-----------|
| (d013) | Output Voltage monitor | 0 to 6000 | - | 0.1v | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the output voltage monitor function (d013). This variable is read only.

| Power | Description | Range of values | Default | Unit | Data size | Attribute |
|--------------|---------------|-----------------|---------|-----------|--------------------|-----------|
| (d014) | Power monitor | 0 to 9999 | - | 0.1 Kw | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the power monitor (d014). This variable is read only.

| RUN-Time | Description | Range of values | Default | Unit | Data size | Attribute |
|-----------------|------------------|-----------------|---------|------|--------------------|-----------|
| (d016) | Run Time monitor | 0 to 999999 | - | Hour | Unsigned 2-word | R |

The data monitored with this variable corresponds to the data monitored by the cumulative operation RUN time monitor (d016). This variable is read only.

| On-Time | Description | Range of values | Default | Unit | Data size | Attribute |
|----------------|-----------------------|-----------------|---------|------|--------------------|-----------|
| (d017) | Power-on time monitor | 0 to 999999 | - | Hour | Unsigned 2-word | R |

The data monitored with this variable corresponds to the data monitored by the cumulative power-on time monitor (d017). This variable is read-only.

| UMon(0) to Umon(2) | Description | Range of values | Default | Unit | Data size | Attribute |
|-----------------------------------|-------------------------------|-------------------------|---------|------|------------------|-----------|
| (d025 to d027) | User Parameter monitor 0 to 2 | -2^{31} to $2^{31}-1$ | 0 | - | Signed 2-word | R/W |

The data monitored with these variables corresponds to the data monitored on d025, d026 and d027. These are monitors available for the user Drive Programming application

| POS | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|--------------------------|--|---------|------|---------------|-----------|
| (d030) | Current Position monitor | $2^{28} - 1$ to $-(2^{28} - 1)$ [$2^{30} - 1$ to $-(2^{30} - 1)$] | - | 1 | Signed 2-word | R |

The data referenced with this variable corresponds to the data monitored by the current position monitor (d030).

With RX when "03" (high-resolution absolute position control) has been selected for control pulse setting (P012), the range in brackets "[]" applies.

| ERR-CNT | Description | Range of values | Default | Unit | Data size | Attribute |
|---------|----------------------|-----------------|---------|-------------|-----------------|-----------|
| (d080) | Trip counter monitor | 0 to 65535 | - | N° of times | Unsigned 1-word | R |

The data monitored with this variable corresponds to the data monitored by the trip counter monitor (d080).

| ERR(1)-ERR(6) | Description | Range of values | Default | Unit | Data size | Attribute |
|---------------|---------------------|-----------------|---------|------|-----------------|-----------|
| (d081-d086) | Trip monitor 1 to 6 | 0 to 127 | - | - | Unsigned 1-word | R |

The data monitored with these variables correspond to the data monitored by trip monitors 1 to 6 (d081 to d086).

| DCV | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|--------------------|-----------------|---------|---------|-----------------|-----------|
| (d102) | DC voltage monitor | 0 to 9999 | - | 0.1 Vdc | Unsigned 1-word | R |

The data referenced with this variable corresponds to the data monitored by the DC voltage monitor (d102).

| STATUS | Description | Range of values | Default | Unit | Data size | Attribute |
|--------|-------------------------|-----------------|---------|------|-----------------|-----------|
| | Inverter status monitor | - | - | - | Unsigned 1-word | R |

This variable can be used to reference inverter status information.

The information is reflected with the following bit weights:

| Bit 9 to 15 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------------|---------------|-------|--------------------------|--------------------------|----------------------|-------|---------|-------|-------|
| Reserve | Under voltage | Reset | Over voltage suppression | Over current suppression | Overload suppression | Retry | Reverse | Trip | Run |

5-4 Terminal Variables

Input/Output Control Instructions

| X(00) - X(09) | Description | Range of Values | Data Size | Attribute |
|---------------|-----------------------------------|-----------------|-----------|-----------|
| | Input terminal 0 to 9 (C011-C017) | 0: Off 1: On | bit | R |

Variables from X(00) to X(06) reflect the multi-functions X(00)–MI1, to X(06)-MI7.
 For MX2, variable X(07) corresponds to the EA input (pulse input) terminal contact digital status.
 Variables from X(08) to X(09) are Extended IO option input contacts when expansion IO board is attached.

| Xw | Description | Range of Values | Data Size | Attribute |
|----|-----------------------|-----------------|--------------------|-----------|
| | Input terminal (word) | 0 to 65535 | Unsigned 1-word | R |

Instruction to access contact inputs by word. Each bit reflects one of the inputs.

| Y(00) - Y(05) | Description | Range of Values | Data Size | Attribute |
|---------------|------------------------|-----------------|-----------|-----------|
| | Output terminal 0 to 5 | 0: Off 1: On | bit | R/W |

Variables from Y(00) to Y(05) write to the multifunction output contacts (C021...C026).
 Y(03) to Y(05) are extended IO option output contacts.

| Yw | Description | Range of Values | Data Size | Attribute |
|----|------------------------|-----------------|--------------------|-----------|
| | Output terminal (word) | 0 to 65535 | Unsigned 1-word | R/W |

This variable can be used to change the digital output terminals in units of word. Each output is one bit.

| XA(0) –XA(2) | Description | Range of Values | Data Size | Attribute |
|--------------|--|-----------------|--------------------------------|-----------|
| XA(0) | General-purpose analog input (O terminal) | 0 to 10000 | Unsigned 1- word (0.01%) | R |
| XA(1) | General-purpose analog input (OI terminal) | | | |
| XA(2) | General-purpose analog input (O2 terminal) only for RX | | | |

These variables can be used to monitor the analog input to the O and OI and O2 terminals. Terminals [O]-[L], [OI]-[L], [O2]-[L]. Associated parameters (A011 to A015, A101 to A105, A111 to A114).

| YA(0) –YA(2) | Description | Range of Values | Data Size | Attribute |
|--------------|--|-----------------|-------------------------------|-----------|
| YA(0) | General-purpose analog output (EO terminal) | 0 to 10000 | Unsigned 1-word (0.01%) | R/W |
| YA(1) | General-purpose analog output (AM terminal) | | | |
| YA(2) | General-purpose analog output (AMI terminal) only for RX | | | |

With this variables we can monitor the analog outputs (any multifunction assigned to them), or write analog output if YA(0) to YA(2) are assigned to analog multifunction parameters (C027, C028 and C029). Value is reflected as a data range from 0% to 100.00%. YA(2) is only available for RX.

| UB(00) – UB(07) | Description | Range of Values | Data Size | Attribute |
|-----------------|------------------------------------|-----------------|-----------|-----------|
| | Internal user contact (bit access) | 0: Off 1: On | bit | R/W |

These variables can be used as bit variable for the user.

| UBw | Description | Range of Values | Data Size | Attribute |
|-----|-------------------------------------|-----------------|--------------------|-----------|
| | Internal user contact (word access) | 0 to 255 | Unsigned 1-word | R/W |

The bit variables reflected as single word.

| TC(0) - TC(7) | Description | Range of Values | Data Size | Attribute |
|---------------|---|-------------------|--------------------|-----------|
| | Timer counters (0 to 7) (Unit: 10ms) | 0 to $2^{31} - 1$ | Unsigned 2-word | R/W |

The timer counters “TC(0)” to “TC(7)” operate as 31-bit-free-running timer counters. They start with the user program startup and are incremented in a 10-ms cycle.

When a timer-start instruction (timer set) or delay operation instruction (delay on or delay off) is executed, the timer counter corresponding to the instruction operates as the counter for output to a specified timer contact. In this case, the counter is cleared to zero when the instruction is executed, start counting, and then stops counting upon reaching the specified count. When a timer-stop instruction (timer off) is executed, the timer counter corresponding to the instruction is cleared to zero and operates as a 31-bit-free-running timer counter that is incremented in a 10-ms cycle.

| TD(0) - TD(7) | Description | Range of Values | Unit | Attribute |
|---------------|---|-----------------|--------------------|-----------|
| | Timer contact output 0 – 7 (bit access) | 0: Off 1: On | Unsigned 1-word | R |

The data in timer contact output variables “TD(0)” to “TD(7)” is changed only when these variables are specified in the timer-start instruction (timer set) or delay operation instruction (delay on or delay off). A timer contact output variable is set to “0”(off) when the counter corresponding to the contact output is cleared to zero, the variable is set to “1”(on) when the counter stops counting (the timing action selected finish).

While a timer counter variable “TC(k)” is being used for a free-running timer counter, timer contact output variable “TD(k)” corresponding to the timer counter variable retains its status.

| TDw | Description | Range of Values | Unit | Attribute |
|-----|------------------------------------|-----------------|--------------------|-----------|
| | Timer contact output (word access) | 0 to 255 | Unsigned 1-word | R |

It access to the timer counter outputs as word.

5-5 Digital input Functions

These variables correspond to the settings available for the digital multifunction input terminals. Setting the variable to 1 will simulate the function as if the terminal was closed in a digital input. It is interesting to note that the multifunction does not need to be configured in order to use the function.

E.g. FW := 1 will generate a RUN Forward command (as used in some examples).

Please refer to the inverter user manual for details about the individual functions.

Values: · 0: Off

· 1: On

| Function | Description | Usage | Comment |
|---------------|--|-------|-------------------|
| FW | Forward | R/W | C001-C007 = 00 |
| RV | Reverse | R/W | C001-C007 = 01 |
| CF1-CF4 | Multi-speed 1-4 | R/W | C001-C007 = 02-05 |
| JG | Jogging | R/W | C001-C007 = 06 |
| DB | External Brake | R/W | C001-C007 = 07 |
| SET | Second control | R/W | C001-C007 = 08 |
| 2CH | 2 nd acceleration/deceleration time | R/W | C001-C007 = 09 |
| FRS | Free run | R/W | C001-C007 = 11 |
| EXT | External trip | R/W | C001-C007 = 12 |
| USP | Unattended start protection | R/W | C001-C007 = 13 |
| CS | Change from commercial power | R/W | C001-C007 = 14 |
| SFT | Software lock | R/W | C001-C007 = 15 |
| AT | Change of analog input | R/W | C001-C007 = 16 |
| SET3 | 3 rd control | R/W | C001-C007 = 17 |
| RS | System reset | R/W | C001-C007 = 18 |
| STA | Start of 3 wires | R/W | C001-C007 = 20 |
| STP | Stop of 3 wires | R/W | C001-C007 = 21 |
| F/R | Forward/Reverse of 3 wires | R/W | C001-C007 = 22 |
| PID | Switch PID | R/W | C001-C007 = 23 |
| PIDC | Reset of PID integration | R/W | C001-C007 = 24 |
| CAS | Control gain switching | R/W | C001-C007 = 26 |
| UP | Increasing speed from remote | R/W | C001-C007 = 27 |
| DWN | Decreasing speed from remote | R/W | C001-C007 = 28 |
| UDC | Clear data from remote operation | R/W | C001-C007 = 29 |
| OPE | Change to operator | R/W | C001-C007 = 31 |
| SF1-SF7 | Multi-speed bit 1-7 | R/W | C001-C007 = 32-38 |
| OLR | Overload protection switch | R/W | C001-C007 = 39 |
| TL | Torque Limit Enable | R/W | C001-C007 = 40 |
| TRQ1-2 | Torque Limit Selection 1-2 | R/W | C001-C007 = 41-42 |
| PPI | P/PI switching | R/W | C001-C007 = 43 |
| BOK | Brake Confirmation | R/W | C001-C007 = 44 |
| ORT | Orientation | R/W | C001-C007 = 45 |
| LAC | LAD Cancel | R/W | C001-C007 = 46 |
| PCLR | Clear Position Deviation | R/W | C001-C007 = 47 |
| STAT | Pulse train position command input permission | R/W | C001-C007 = 48 |
| ADD | Add Setting Frequency | R/W | C001-C007 = 50 |
| F-TM | Forced Terminal Block | R/W | C001-C007 = 51 |
| ATR | Torque reference input permission | R/W | C001-C007 = 52 |
| KHC | Integrated power clear | R/W | C001-C007 = 53 |
| SON | Servo ON | R/W | C001-C007 = 54 |
| FOC | Preliminary excitation | R/W | C001-C007 = 55 |
| X(00) – X(07) | Drive Programming (MI1-MI8) | R/W | C001-C007 = 56-63 |
| AHD | Analog command on hold | R/W | C001-C007 = 65 |
| CP1-3 | Position command selection 1-3 | R/W | C001-C007 = 66-68 |
| ORL | Origin return limit signal | R/W | C001-C007 = 69 |
| ORG | Origin return start signal | R/W | C001-C007 = 70 |
| FOT | Forward driving stop | R/W | C001-C007 = 71 |
| ROT | Reverse driving stop | R/W | C001-C007 = 72 |
| SPD | Speed/Position switching | R/W | C001-C007 = 73 |

| Function | Description | Usage | Comment |
|----------|---|-------|----------------|
| PCNT | Pulse counter | R/W | C001-C007 = 74 |
| PCC | Pulse counter clear | R/W | C001-C007 = 75 |
| RS485 | Inverter communication start terminal | R/W | C001-C007 = 81 |
| HLD | HOLD Acceleration / deceleration stopping | R/W | C001-C007 = 83 |
| ROK | Operation OK signal | R/W | C001-C007 = 84 |
| DISP | Display limitation terminal | R/W | C001-C007 = 86 |

5-6 Digital Output Functions

These variables correspond to the settings available for the digital multifunction output terminals. The variable can read and used as it would be for an external device connected to the digital output configured for the function.


It is interesting to note that digital outputs are not required to be assigned in order to use the function within the program (in other words, no waste of digital outputs required).

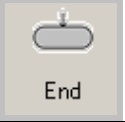
| | | | |
|-------|-----------------------------------|---|-----------------------|
| RUN | Running | R | C021, C022, C026 = 00 |
| FA1 | Reaching constant speed | R | C021, C022, C026 = 01 |
| FA2 | Greater than setting frequency | R | C021, C022, C026 = 02 |
| OL | Overload preannounce | R | C021, C022, C026 = 03 |
| OD | PID deviation overrate | R | C021, C022, C026 = 04 |
| AL | Trip signal | R | C021, C022, C026 = 05 |
| FA3 | Only the setting frequency | R | C021, C022, C026 = 06 |
| OTQ | Over torque/under torque | R | C021, C022, C026 = 07 |
| UV | Under voltage signal | R | C021, C022, C026 = 09 |
| TRQ | Torque limitation signal | R | C021, C022, C026 = 10 |
| RNT | RUN time over | R | C021, C022, C026 = 11 |
| ONT | ON time over | R | C021, C022, C026 = 12 |
| THM | Thermal warning | R | C021, C022, C026 = 13 |
| BRK | Brake open | R | C021, C022, C026 = 19 |
| BER | Brake error | R | C021, C022, C026 = 20 |
| ZS | Zero speed signal | R | C021, C022, C026 = 21 |
| DSE | Speed deviation overrate | R | C021, C022, C026 = 22 |
| POK | Positioning operation complete | R | C021, C022, C026 = 23 |
| FA4 | Greater than setting frequency 2 | R | C021, C022, C026 = 24 |
| FA5 | Only the setting frequency 2 | R | C021, C022, C026 = 25 |
| OL2 | Overload preannounce 2 | R | C021, C022, C026 = 26 |
| ODc | Analog O break detection | R | C021, C022, C026 = 27 |
| OIDc | Analog OI break detection | R | C021, C022, C026 = 28 |
| FBV | PID feedback comparison | R | C021, C022, C026 = 31 |
| NDc | Communication break detection | R | C021, C022, C026 = 32 |
| LOG1 | Result of logic operation 1 | R | C021, C022, C026 = 33 |
| LOG 2 | Result of logic operation 2 | R | C021, C022, C026 = 34 |
| LOG 3 | Result of logic operation 3 | R | C021, C022, C026 = 35 |
| LOG 4 | Result of logic operation 4 | R | C021, C022, C026 = 36 |
| LOG 5 | Result of logic operation 5 | R | C021, C022, C026 = 37 |
| LOG 6 | Result of logic operation 6 | R | C021, C022, C026 = 38 |
| WAC | Condenser life-span preannounce | R | C021, C022, C026 = 39 |
| WAF | Fan life-span preannounce | R | C021, C022, C026 = 40 |
| FR | Start contact signal | R | C021, C022, C026 = 41 |
| OHF | Cooling fan over heat preannounce | R | C021, C022, C026 = 42 |
| LOC | Low electricity signal | R | C021, C022, C026 = 43 |
| Y(00) | Drive Programming (MO1) | R | C021, C022, C026 = 44 |
| Y(01) | Drive Programming (MO2) | R | C021, C022, C026 = 45 |
| Y(02) | Drive Programming (MO3) | R | C021, C022, C026 = 46 |
| Y(03) | Drive Programming (MO4) | R | C021, C022, C026 = 47 |
| Y(04) | Drive Programming (MO5) | R | C021, C022, C026 = 48 |
| Y(05) | Drive Programming (MO6) | R | C021, C022, C026 = 49 |
| IRDY | Operation setup complete | R | C021, C022, C026 = 50 |
| FWR | Forward running signal | R | C021, C022, C026 = 51 |


| | | | |
|------|---------------------------------|---|-----------------------|
| RVR | Reverse running signal | R | C021, C022, C026 = 52 |
| MJA | Serious failure signal | R | C021, C022, C026 = 53 |
| WCO | Window comparator O | R | C021, C022, C026 = 54 |
| WCOI | Window comparator OI | R | C021, C022, C026 = 55 |
| WCO2 | Window comparator O2 | R | C021, C022, C026 = 56 |
| FREF | Command frequency selected mode | R | C021, C022, C026 = 58 |
| REF | Command operation mode | R | C021, C022, C026 = 59 |
| SETM | Setting motor | R | C021, C022, C026 = 60 |
| EDM | STO operation monitor signal | R | C021, C022, C026 = 62 |


6- Drive Programming Instructions


6-1 Control Commands

| Entry | | |
|--|---|-----------|
| Command | Description | Arguments |
|  Entry | It indicates the beginning of the task. | --- |
| Format | | |
| --- | | |
| Note: It is compulsory to have Entry at the beginning of each task. | | |

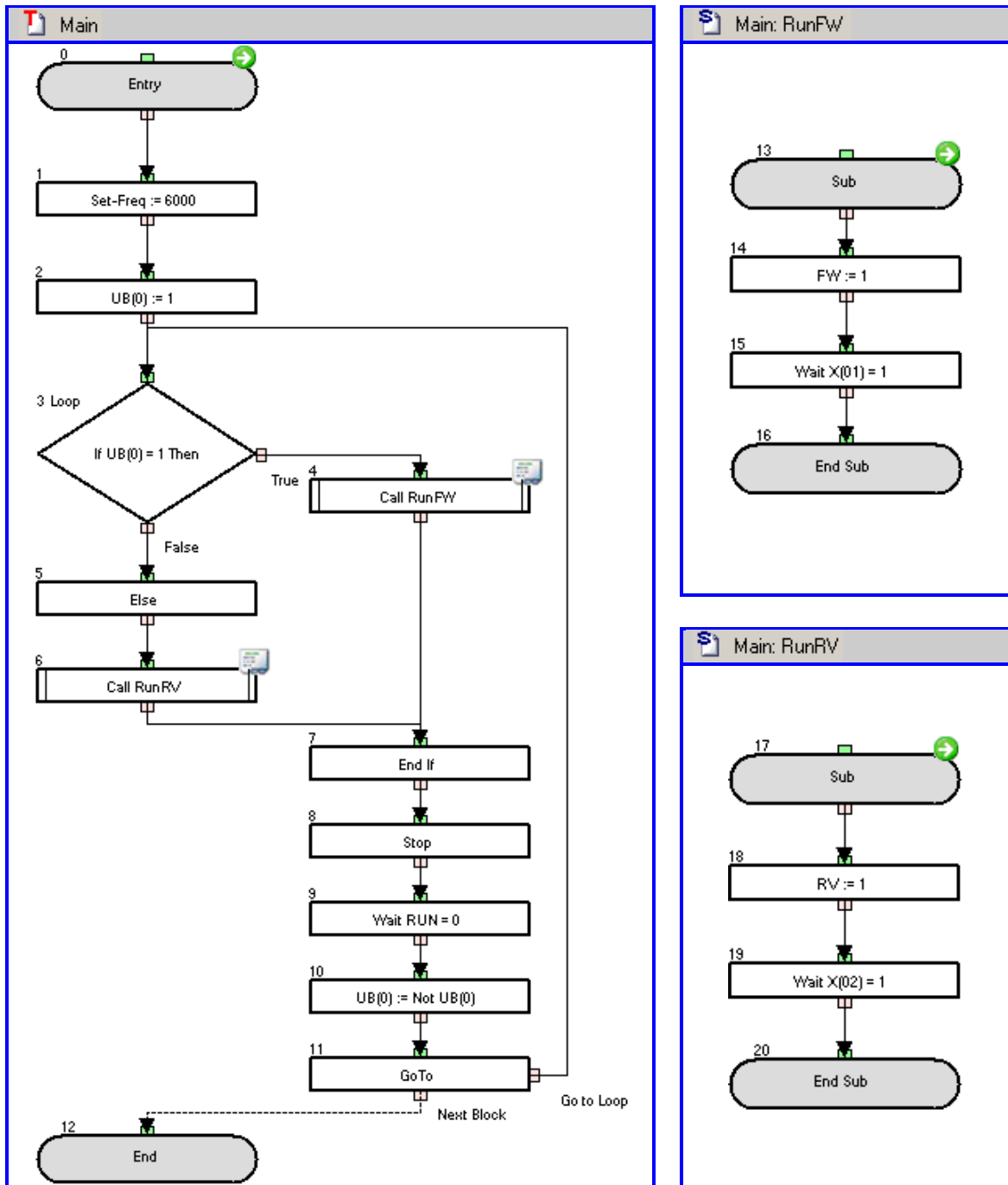
| End | | |
|--|-----------------------------------|-----------|
| Command | Description | Arguments |
|  End | It indicates the end of the task. | --- |
| Format | | |
| --- | | |
| Note: It is compulsory to have End at the end of each task. | | |

| Call | | |
|--|--------------------------|---|
| Command | Description | Arguments |
|  Call | It jumps to a subroutine | <ul style="list-style-type: none"> • Subroutine: Subroutines are identified by a name or alias defined by the user. |
| Format | | |
| call <subroutine> | | |
| Note: After the execution of the subroutine ends, the next instruction line after the call is executed. | | |


| Sub | | |
|--|---|-----------|
| Command | Description | Arguments |
|  Sub | It indicates the beginning of the subroutine. | --- |
| Format | | |
| --- | | |
| Note: It is compulsory to have Sub at the beginning of each subroutine. | | |

| End Sub | | |
|---|---------------------------------------|-----------|
| Command | Description | Arguments |
|  EndSub | It indicates the end of a subroutine. | --- |
| Format | | |
| --- | | |
| Note: It is compulsory to have End Sub at the end of each subroutine. | | |

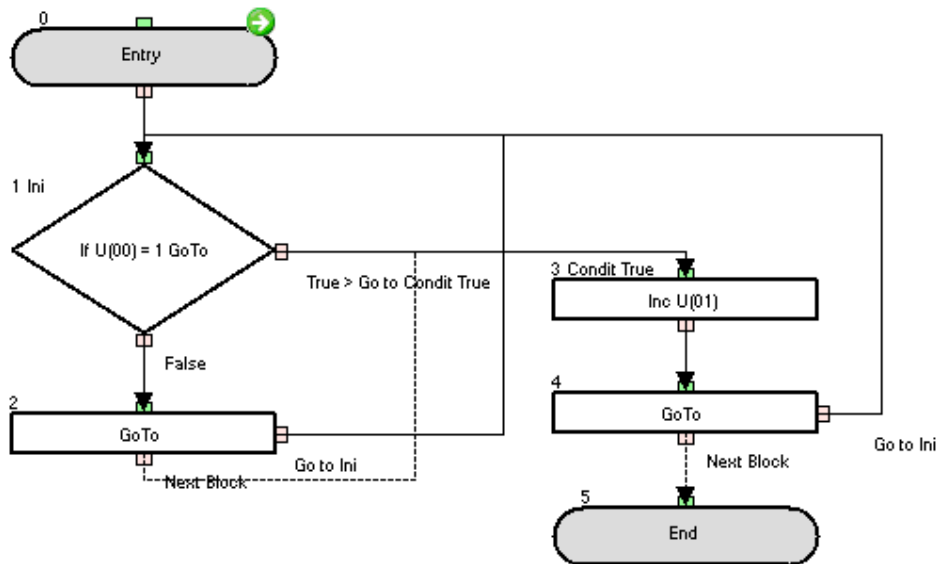
Example




A forward and reverse run at 60Hz is repeated continuously between two limits X(01) and X(02).

| Go To | | |
|---|--|--|
| Command | Description | Arguments |
|  | Use this instruction to branch processing unconditionally to the step labeled with label name. | <ul style="list-style-type: none"> • Label: A name that is used to identify a particular function block in the task. |
| Format | | |
| GoTo <label> | | |
| Note: The instruction must also be connected to the next program block you want to be executed. This is necessary to make clear the flow of the program. | | |

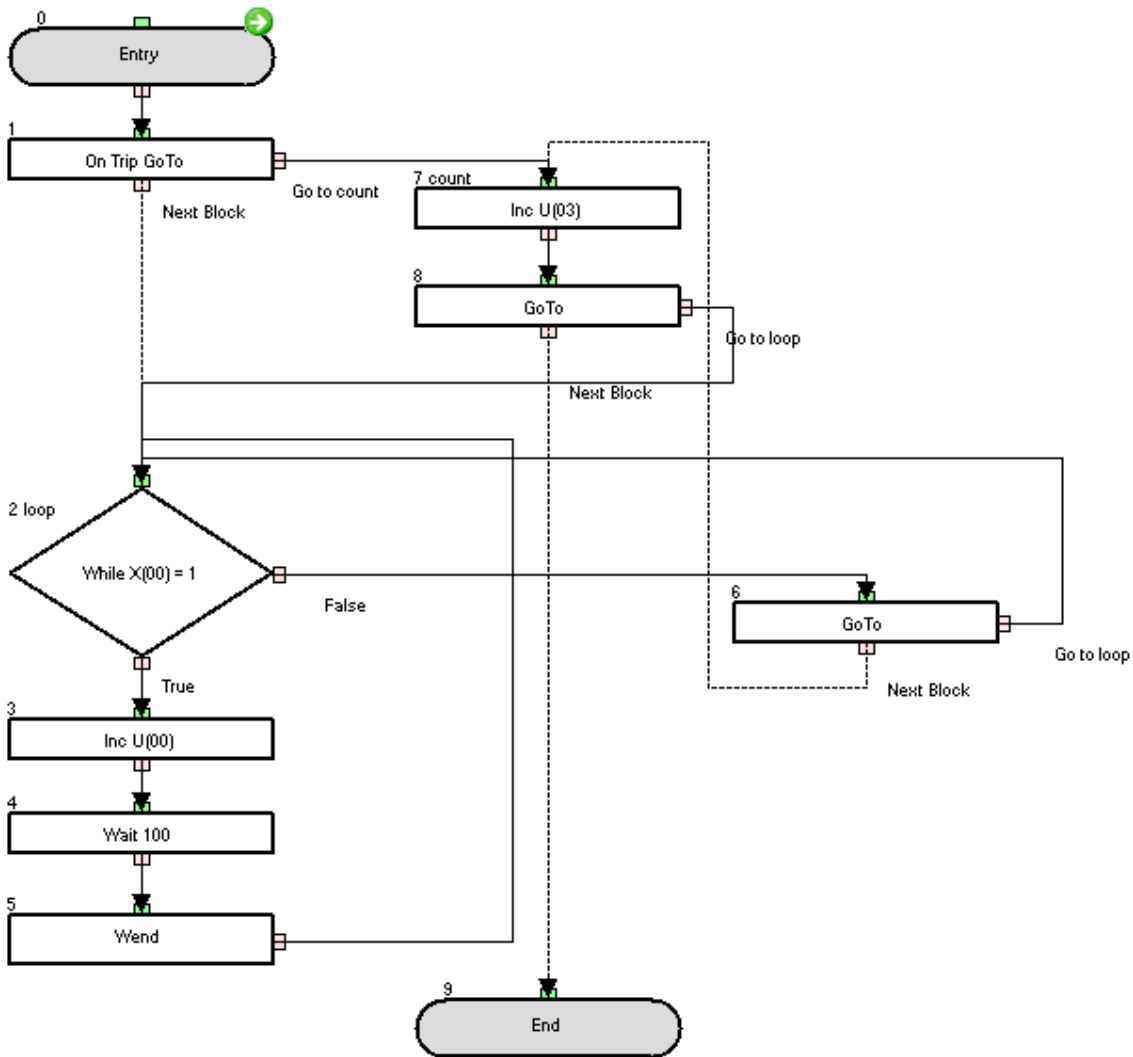
Example:




Change parameter P100 in order to test the **GoTo** function with this sample. When P100=1, P101 starts counting. When P100<>1 stops counting.

| On Trip | | |
|---|---|--|
| Command | Description | Arguments |
|  | This instruction makes conditional branching in case a trip in the inverter occurs. | <ul style="list-style-type: none"> • Label: A name that is used to identify a particular function block in the task. |
| Format | | |
| On Trip goto <label> | | |
| Note: The On Trip instruction works as a trigger arming. The instruction is executed once, after, at any moment a trip occurs, the program immediately jumps to the designed label and the On trip trigger is disarmed. | | |

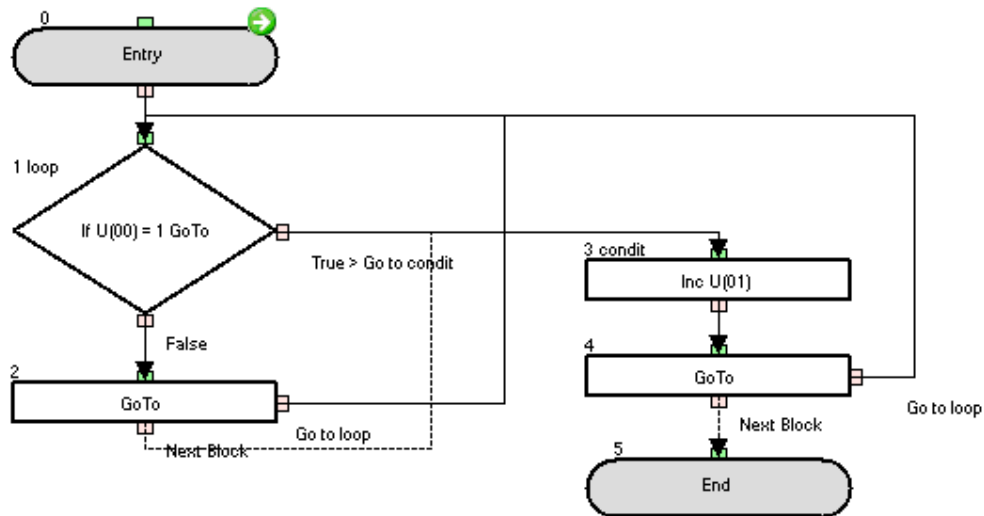
Example





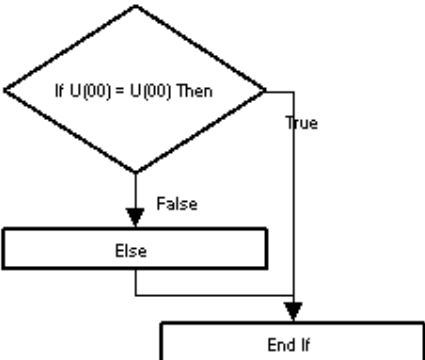
When the digital input is set to ON value, then P100 parameter is incrementing every second. If a trip is generated (like by external trip input) then P103 increments count. And then goes to the beginning of the task.

| If | | |
|---|--|--|
| Command | Description | Arguments |
|  | Jump to a label when a condition is satisfied. | <ul style="list-style-type: none"> • Condition: A comparison between two variables or constant with the format <Left hand value><Comparison><Right Hand Value> -Left hand value: any variable or constant(range -128 to 127) -Comparison: =, <, >, <=, >=, <> -Right hand value: any variable or constant(range -128 to 127) • Label: A name that is used to identify a particular function block in the task. |
| Format | | |
| If <condition> GoTo <label> | | |

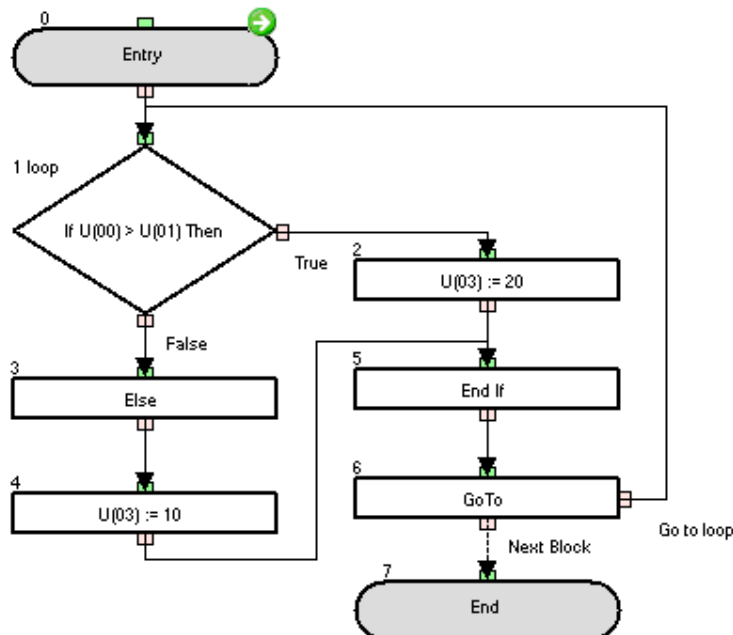
Example:







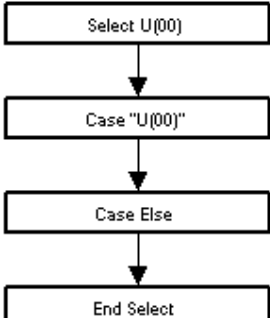
Change parameter P100 in order to test the **GoTo** function with this sample. When P100 = 1, P101 starts counting. When P100 <> 1 stops counting.

| Ifs/ Else / End If | | |
|--|---|---|
| Command | Description | Arguments |
|  Ifs  Ifs ... | This instruction executes different portion of code based on a condition. When the condition is met, this instruction executes <instruction set 1>. When the condition is not met, this instruction executes <instruction set 2>. | <ul style="list-style-type: none"> • Condition: A comparison between two variables or constant with the format <Left hand Value><Comparison><Right hand Value> -<u>Left hand value:</u> any variable or constant (range -128 to 127) -<u>Comparison:</u> =, <, >, <=, >=, <> -<u>Right hand value:</u> any variable or constant (range -128 to 127) • Instruction set 1: One or more instructions, until Else instruction. It can contain nested instructions (up to 8 level of nesting) • Instruction set 2: One or more instructions, until End If instruction. It can contain nested instructions (up to 8 level of nesting). |
| Format | | |
| Ifs <condition> Then <instruction set 1> Else <instruction set 2> Endif | |  |

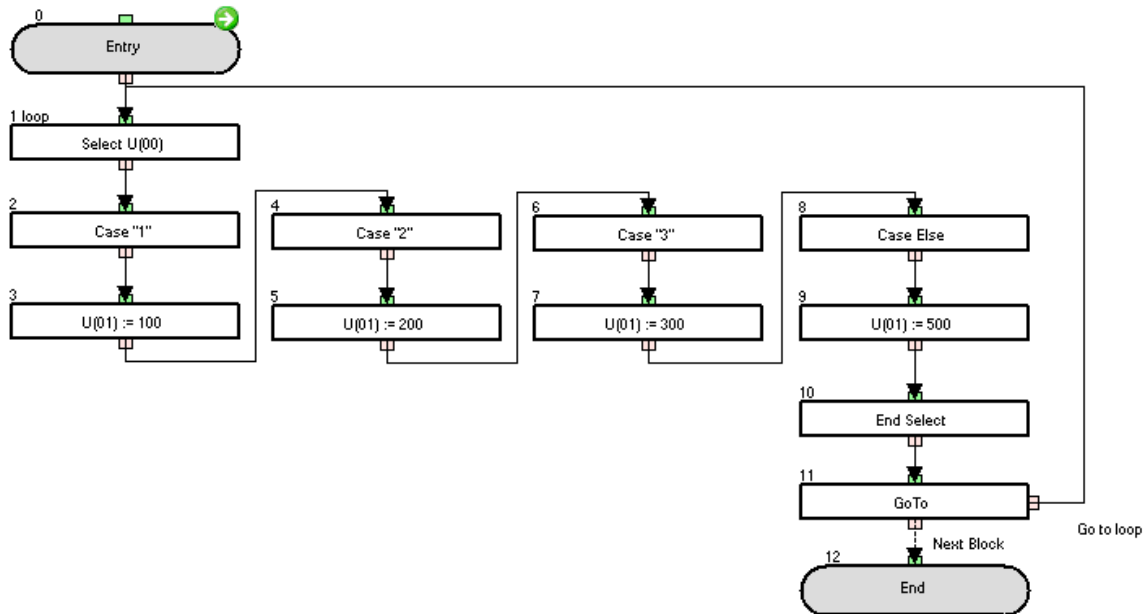
Example




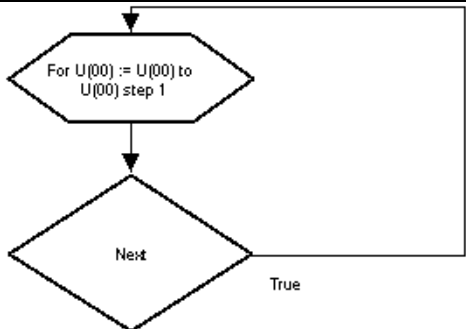
The example changes the value of P103 based on the value of parameter P100 and P101. If P100 is bigger than P101 then P103=10. If not P103=20.

| Select / Case / End Select | | |
|--|--|---|
| Command | Description | Arguments |
|  Select | This instruction allows multiple program sections to be executed depending on a variable value. It For a particular CASE section it Executes <instruction set n> when <conditional variable> matches <conditional value n> If no <conditional variable> don't match any of the CASE section, the <instruction set if no other> (Case Else) is executed. This instruction is convenient when multiple choices have to be done from parameter value. It makes many if / then structures simpler. This instruction is recommended to organize our program by using subroutine calls as instruction set. | <ul style="list-style-type: none"> • Conditional variable: the select variable to the instruction. • Conditional value x: the value of the variable. • Instruction set x: One or more instructions, until next case or end select. It can contain nested instructions (up to 8 level of nesting). |
|  Case | | |
|  CaseElse | | |
|  EndSelect | | |
| Format | | |
| Select <conditional variable> Case <conditional value 1> <instruction set 1> Case <conditional value 2> <instruction set 2> ... Case Else <instruction set if no other> End select |  | |

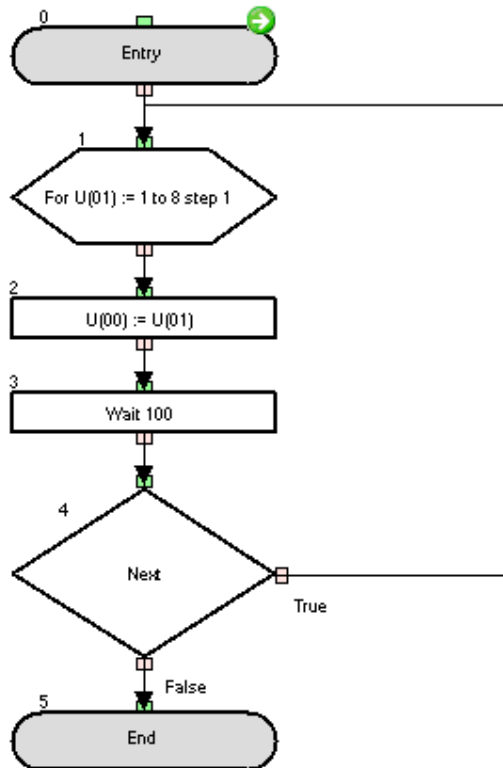
Example




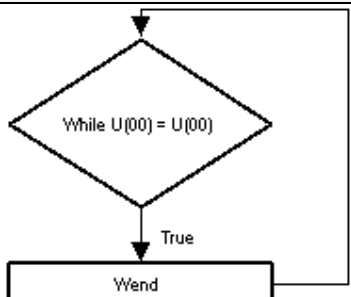
The P101 parameter is set to 100, 200, 300 or 500 depending on the value of the P100 parameter (1, 2, 3 or any other, respectively).

| For / Next | | |
|---|---|---|
| Command | Description | Arguments |
|  | <p>Executes <instruction set> repeatedly until <variable> reaches <end value> that is added <incremental value> each cycle.</p> | <ul style="list-style-type: none"> • Variable: any variable • Start value: Initial value, it is the value assigned to the variable in the first loop. • End value: Value than processing exits the loop. • Incremental value: The variable will be incremented by this value in each loop. • Instruction set: One or more instructions, until Next instruction. It can contain nested instructions (up to 8 level of nesting). |
| Format | | |
| <p>For <variable><start value><end value><incremental value> <instruction set> Next</p> |  | |

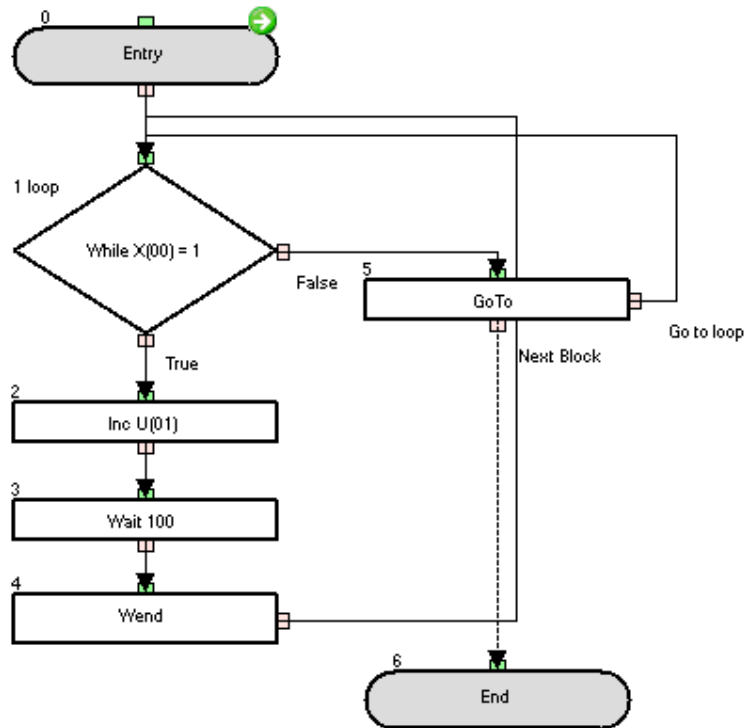
Example




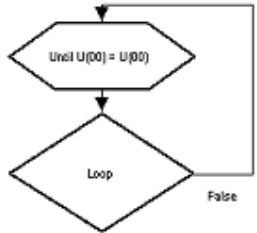
This example make the variable U(00) P(100) count from 1 to 8 each second.

| While / Wend | | |
|---|--|---|
| Command | Description | Arguments |
|  | Executes <instruction set> while a condition is met. | <ul style="list-style-type: none"> • Condition: A comparison between two variables or constants with the format <Left hand value><Comparison><Right Hand Value> <ul style="list-style-type: none"> - <u>Left hand value:</u> any variable or constant (range -128 to 127). - <u>Comparison:</u> =, <, >, <=, >=, <> - <u>Right hand value:</u> any variable or constant (range -128 to 127). • Instruction set: One or more instructions, until Wend instruction. It can contain nested instructions (up to 8 level of nesting). |
| Format | | |
| While <condition> <instruction set> Wend | |  |

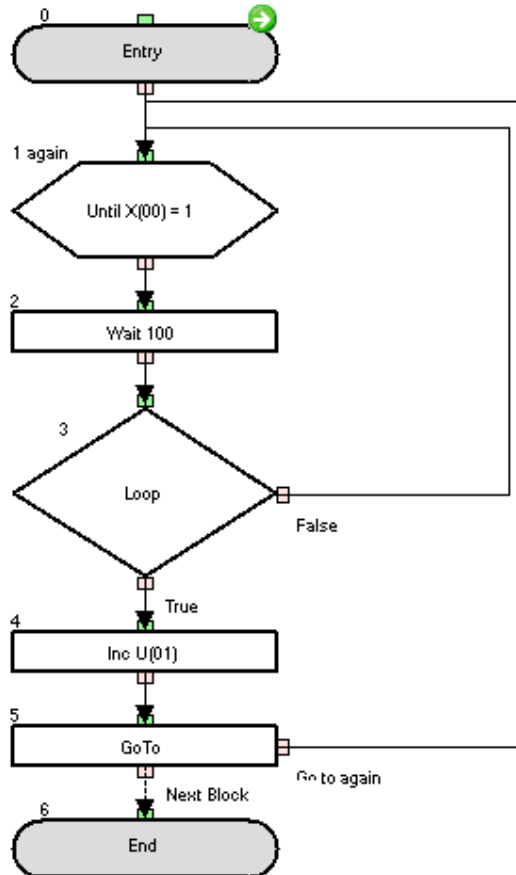
Example





The code will increment P101 parameter every second while the digital input X(00) is closed (**while-wend** loop). If it is open, P101 is not increased (**GoTo-label loop** loop; the **while – wend** portion is not executed). Digital input has to be configured in the multifunction input.

| Until / Loop | | |
|--|---|--|
| Command | Description | Arguments |
|  <p>Until</p> | <p>Executes <instruction set> until a <condition> is met.</p> | <ul style="list-style-type: none"> • Condition: A comparison between two variables or constants with the format <Left hand value><Comparison><Right Hand Value> - Left hand value: any variable or constant (range -128 to 127) - Comparison: =, <, >, <=, >=, <> - Right hand value: any variable or constant (range -128 to 127) • Instruction set: One or more instructions, until Loop instruction. It can contain nested instructions (up to 8 level of nesting) |
| Format | | |
| <p>Until <condition> <instruction set> Loop</p> | |  |

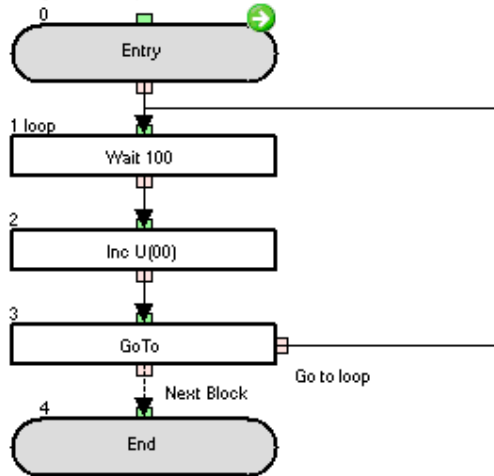
Example



This code will increment while the digital input is closed. If it is open, then it will stay in the **until-loop** portion. The check of the input is every second because of this structure. Digital input has to be configured in the multifunction input.

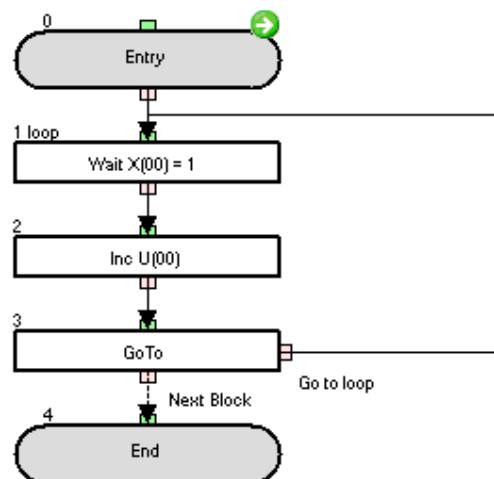
| Wait | | |
|--|--|--|
| Command | Description | Arguments |
|  WaitTime  WaitCond | This instruction makes the program wait for a number of seconds or until a condition is met. | <ul style="list-style-type: none"> Value: any variable or constant (time in 10 x ms). Condition: A comparison between two variables or constant with the format <Left hand value><Comparison><Right Hand Value> <ul style="list-style-type: none"> -Wait Time value from 0 to 32767 * 10ms -Left hand value: any variable or constant (range -128 to 127) -Comparison: =, <, >, <=, >=, <>. -Right hand value: any variable or constant (range -128 to 127) |
| Format | | |
| Wait <value> or <condition> NOTE: WaitTime is not accurate way to measure time, please use internal timers or external Real Time Clock in LCD for accurate time measurement. | | |

Example Wait Time: wait during a time period.




The P100 parameter is increased every second.

Example Wait condition: wait for condition.

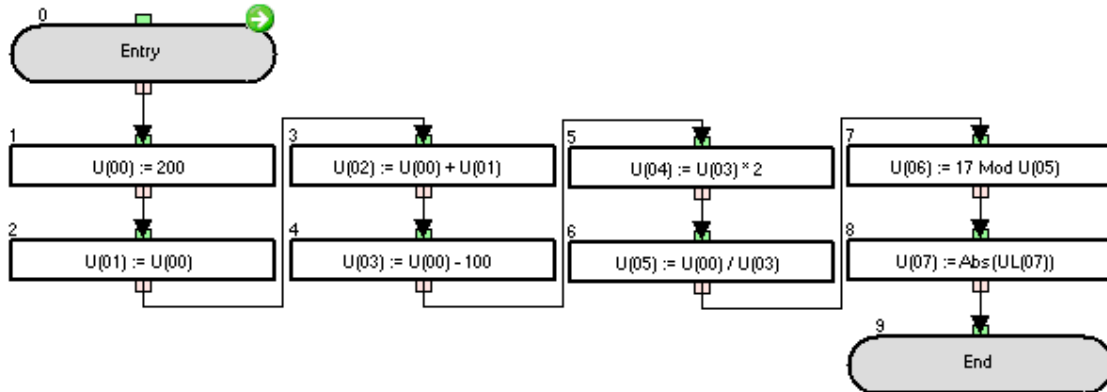


The program waits until the digital input is closed (you need to set one of the multifunction inputs for this), and then P100 parameter is increased.


6-2 Arithmetic and Logic Commands

| = (Substitution) | | |
|--|------------------------------|---|
| Command | Description | Arguments |
|  | Assigns <value> to <result>. | <ul style="list-style-type: none"> • Result: any variable. • Value: any variable or constant (range -2147483648 to 2147483647). |
| Format | | |
| <result> = <value> | | |
| Warning: Drive programming does not control overflow/underflow. The application should take care. | | |

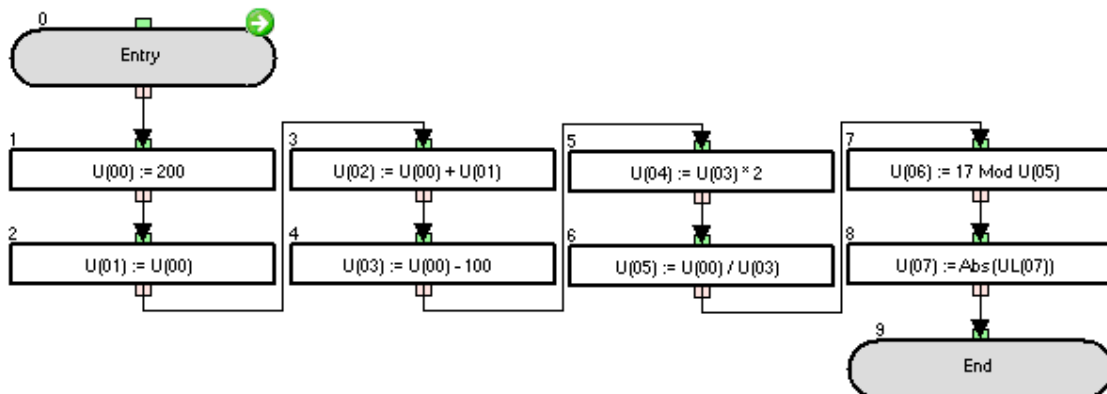
Example:




The P100 and P101 parameters are set to 200.

| + (Addition) | | |
|--|-------------------------------|---|
| Command | Description | Arguments |
|  | Adds <value 1> and <value 2>. | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127) • Value 2: any variable or constant (range -2147483648 to 2147483647). |
| Format | | |
| <result> = <value 1> + <value 2> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

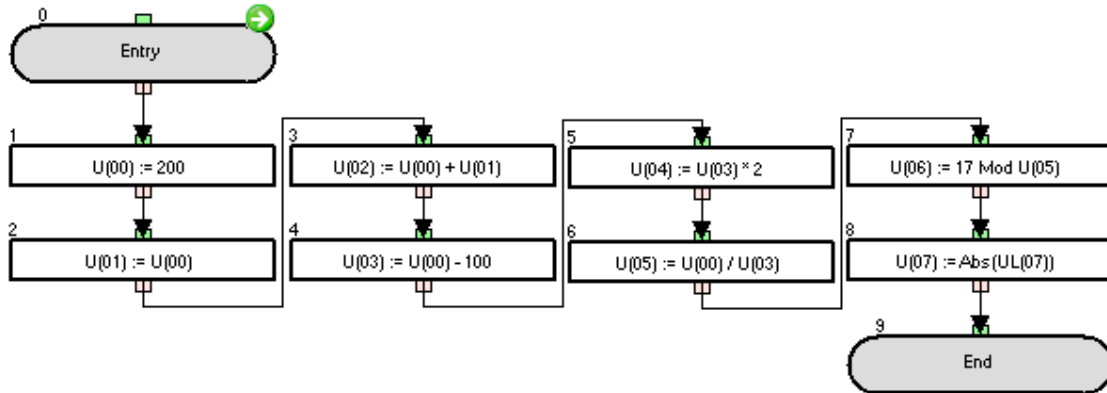
Example:




The P102 parameter calculation result is 400.

| - (Subtraction) | | |
|--|-------------------------------------|--|
| Command | Description | Arguments |
|  | Subtracts <value 2> from <value 1>. | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127). • Value 2: any variable or constant (range -2147483648 to 2147483647). |
| Format | | |
| <result>= <value 1> - <value 2> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

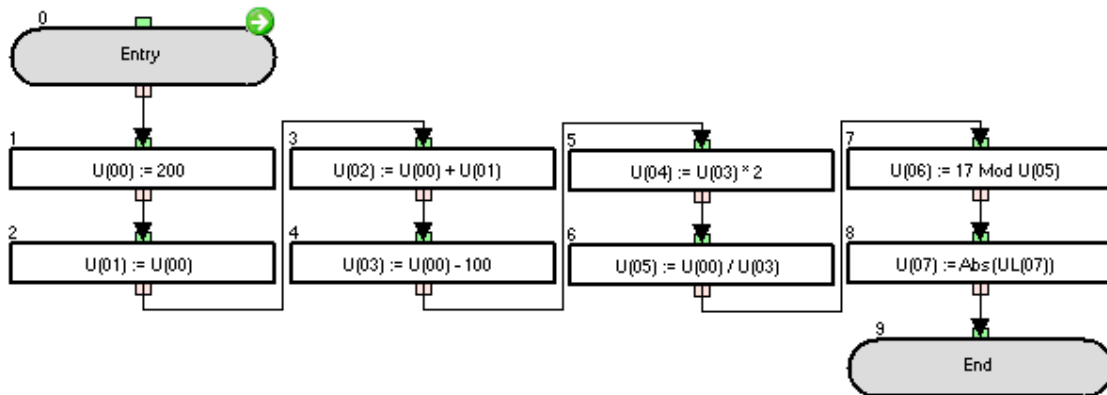
Example:




The P103 parameter calculation result is 100.

| *(Multiplication) | | |
|--|------------------------------------|--|
| Command | Description | Arguments |
|  | Multiplies <value 1> by <value 2>. | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127). • Value 2: any variable or constant (range -2147483648 to 2147483647). |
| Format | | |
| <result> = <value 1> * <value 2> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

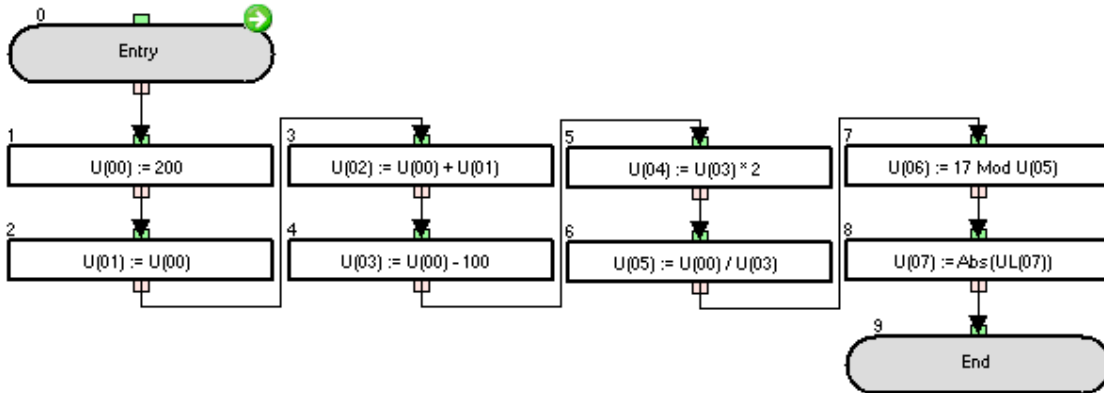
Example:



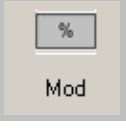
The P104 parameter is set to 200.

| /(Division) | | |
|--|---------------------------------|---|
| Command | Description | Arguments |
|  | Divides <value 1> by <value 2>. | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127) • Value 2: any variable or constant (range -2147483648 to 2147483647). |
| Format | | |
| <result> = <value 1> / <value 2> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

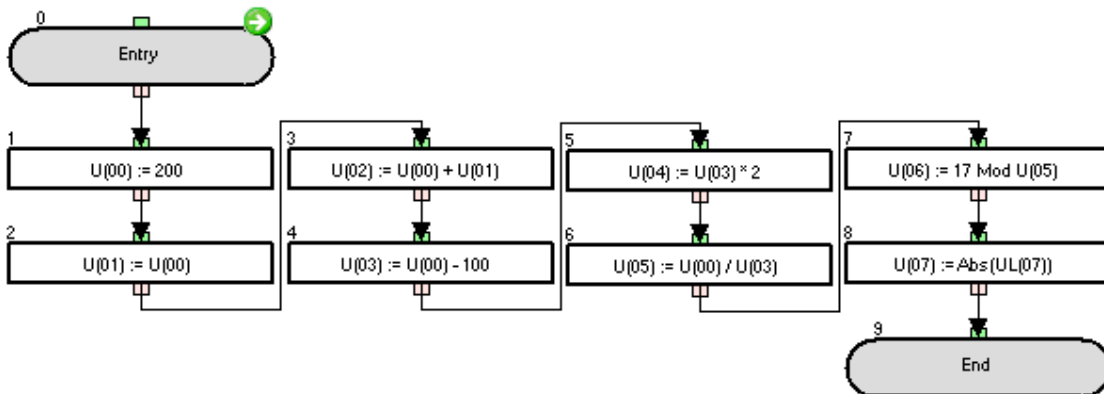
Example:



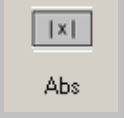
The P105 parameter calculation result is 2.

| % (Mod) | | |
|--|------------------------|---|
| Command | Description | Arguments |
|  | Remainder of division. | <ul style="list-style-type: none"> • Result: Any variable. • Value 1: any variable or constant (range -128 to 127). • Value 2: any variable or constant (range -2147483648 to 2147483647) |
| Format | | |
| <result> = <value 1> Mod <value 2> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

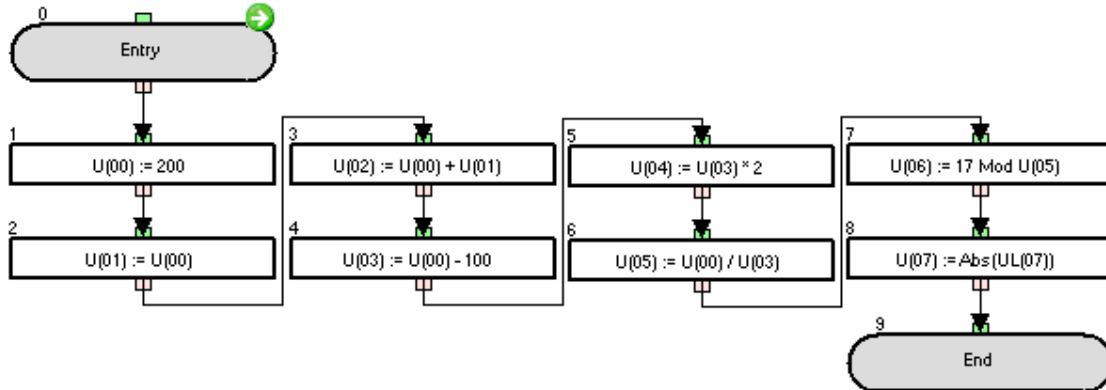
Example:




The P106 parameter calculation result is 1.

| Abs | | |
|--|-----------------|---|
| Command | Description | Arguments |
|  | Absolute value. | <ul style="list-style-type: none"> • Result: any variable. • Value: any variable or constant (range -2147483648 to 2147483647). |
| Format | | |
| <result> = Abs <value> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

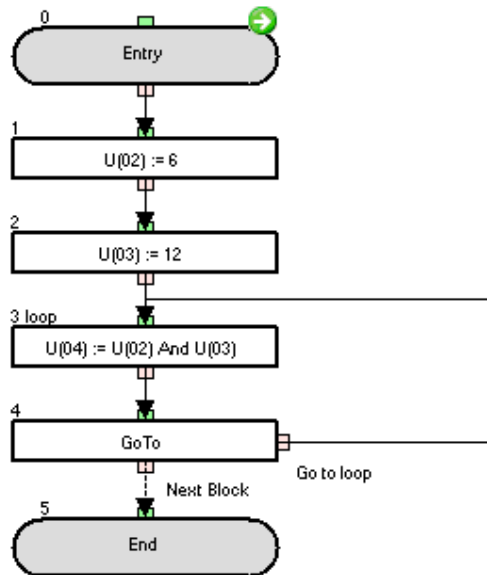
Example:




If the UL(07) variable is set to a positive value in the program variables list, then P107=UL(07). If the UL(07) is set to a negative value, then P107=-UL(07).

| And | | | | |
|--|------------------------|----------------|--|---------------|
| Command | Description | | Arguments | |
|  | And (logical product). | | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127). • Value 2: any variable or constant (range -2147483648 to 2147483647). | |
| | Value 1 | Value 2 | | Result |
| | 0 | 0 | | 0 |
| | 0 | 1 | | 0 |
| | 1 | 0 | | 0 |
| 1 | 1 | 1 | | |
| Format | | | | |
| <result> = <value 1> And <Value 2> | | | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | | | |

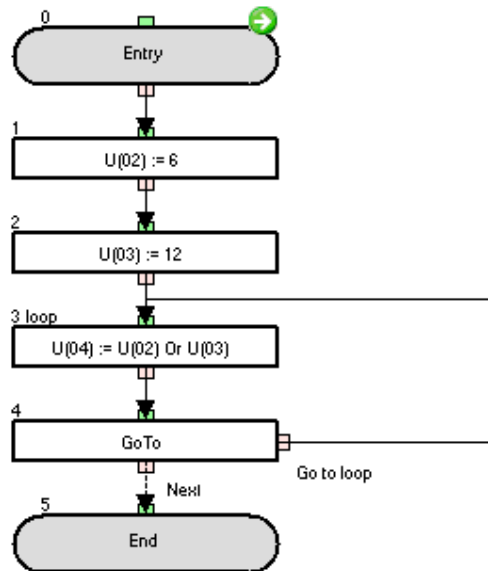
Example



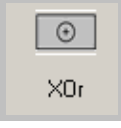
The initial P104 parameter calculation result is 4, as 6 in binary format is 00000110 and 12 in binary format is 00001100, so the result of the **and** operation is 00000100 that is 4 in decimal format. If P102 and P103 are changed by the user, then P104 will recalculate accordingly.

| Or | | | | |
|--|------------------------|----------------|---------------|--|
| Command | Description | | | Arguments |
|  | Or (logical addition). | | | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127). • Value 2: any variable or constant (range -2147483648 to 2147483647). |
| | Value 1 | Value 2 | Result | |
| | 0 | 0 | 0 | |
| | 0 | 1 | 1 | |
| | 1 | 0 | 1 | |
| | 1 | 1 | 1 | |
| Format | | | | |
| <result> = <value 1> Or <value 2> | | | | |
| Warning: Drive programming does not control overflow/underflow. The application should take care. | | | | |

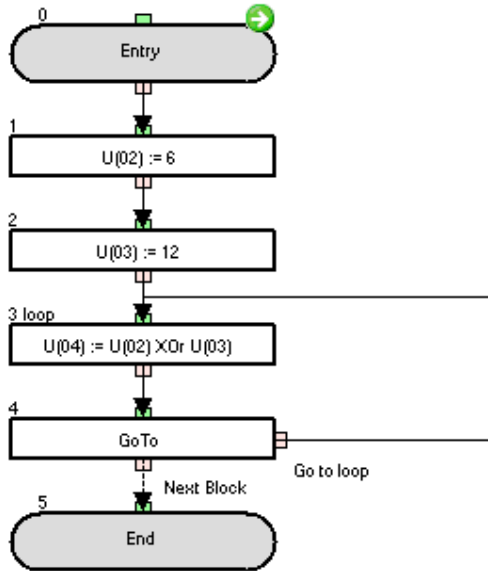
Example




The initial P104 parameter calculation result is 14, as 6 in binary format is 00000110 and 12 in binary format is 00001100, so the result of the operation is 00001110 that is 14 in decimal format. If P102 and P103 are changed by the user, then P104 will recalculate accordingly.

| XOr | | | | |
|--|-------------------|----------------|---------------|--|
| Command | Description | | | Arguments |
|  | XOr(exclusive-or) | | | <ul style="list-style-type: none"> • Result: any variable. • Value 1: any variable or constant (range -128 to 127). • Value 2: any variable or constant (range -2147483648 to 2147483647). |
| | Value 1 | Value 2 | Result | |
| | 0 | 0 | 0 | |
| | 0 | 1 | 1 | |
| | 1 | 1 | 0 | |
| Format | | | | |
| <result>= <value 1> XOr <value 2> | | | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | | | |

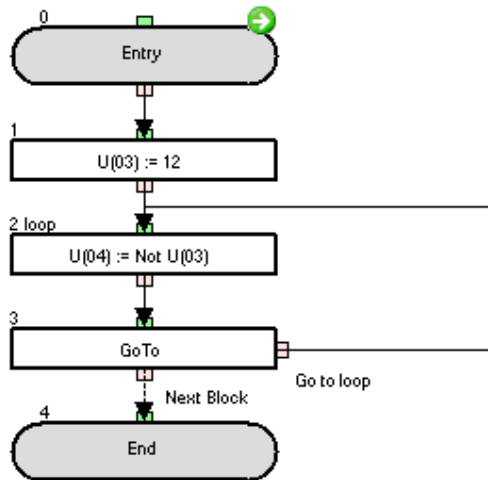
Example




The initial P104 parameter calculation result is 10, as 6 in binary format is 00000110 and 12 in binary format is 00001100, so the result of the **XOr** operation is 00001010 that is 10 in decimal format. If P102 and P103 are changed by the user, then P104 will recalculate accordingly.

| Not | | | | | | | | |
|--|---|-----------|--------|---|---|---|---|--|
| Command | Description | Arguments | | | | | | |
|  | Not (negation) <table border="1"> <thead> <tr> <th>Value 1</th> <th>Result</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </tbody> </table> | Value 1 | Result | 0 | 1 | 1 | 0 | <ul style="list-style-type: none"> • Result: any variable, except variables with bit data size (Note 1) • Value: any variable or constant, except variables with bit data size (Note 1) (range -2147483648 to 2147483647). |
| Value 1 | Result | | | | | | | |
| 0 | 1 | | | | | | | |
| 1 | 0 | | | | | | | |
| Format | | | | | | | | |
| <result> = Not<value> | | | | | | | | |
| <p>Note: Unexpected result will be obtained with instructions like UB(1) = Not UB(0). Please use <i>XOr</i> command to negate variables with bit data size in Drive Programming as shown on the next examples:</p> <ul style="list-style-type: none"> • Example 1: UB(1) = UB(0) Xor 1 • Example 2: UB(2) = X(00) Xor 1 | | | | | | | | |
| <p>Warning: Drive Programming does not control overflow/underflow. The application should take care.</p> | | | | | | | | |

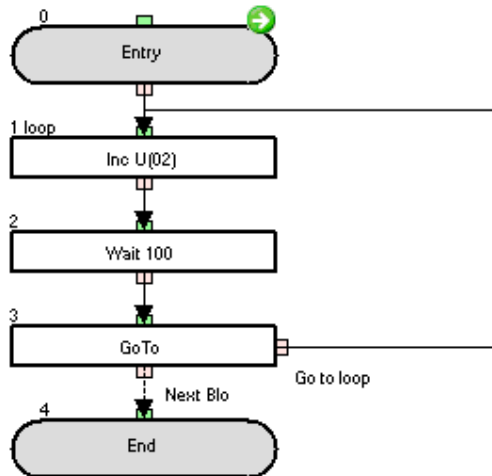
Example:




The initial P104 parameter calculation result is 249, as 6 in binary format is 00000110, so the result of the **not** operation is 11111001 that is 249 in decimal format. If P103 is changed by the user, then P104 will recalculate accordingly.

| Inc | | |
|--|-------------------------|---|
| Command | Description | Arguments |
|  | Increases a value by 1. | <ul style="list-style-type: none"> • Value: any variable. |
| Format | | |
| Inc<value> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |

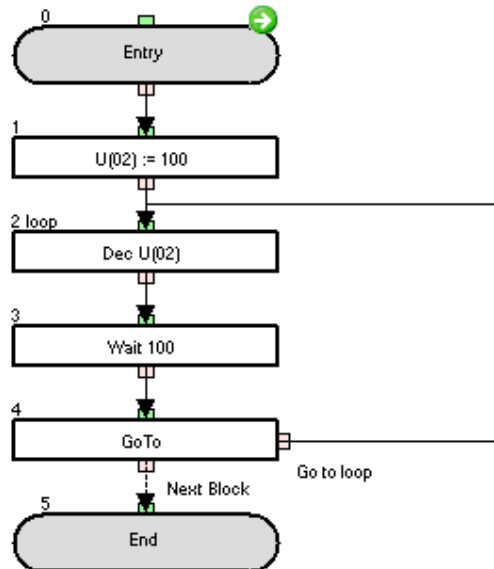
Example:



The P102 parameter is incremented by 1 every second.

| Dec | | |
|--|--------------------------|---|
| Command | Description | Arguments |
|  | Decrements a value by 1. | <ul style="list-style-type: none"> • Value: any variable. |
| Format | | |
| Dec<value> | | |
| Warning: Drive Programming does not control overflow/underflow. The application should take care. | | |


Example:



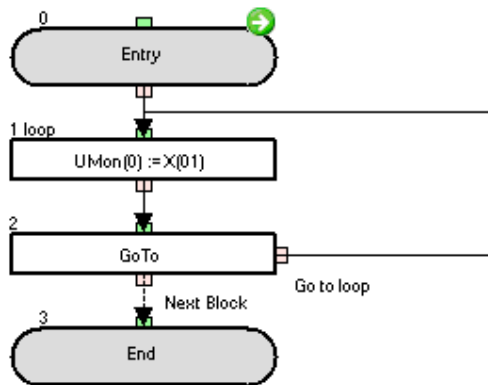
The P102 parameter is decremented by 1 every second.

6-3 Input/Output Control Commands


To improve the memory optimization, use Input/Output Control Commands (4 bytes) instead of the Equal Arithmetic Command "=" (8bytes).

| Var = X(i) | | |
|---|---|--|
| Command | Description | Arguments |
|  | Instruction to access contact inputs. Reflects the state of the input. | <ul style="list-style-type: none"> • Variable: any variable (the value of the variable will be 0 or 1). • i: Number of the contact input (range 0 to 9). |
| Format | | |
| <variable>=X(i) | | |
| <p>Note: The inputs have to be assigned to digital multifunction input (by the multifunction 56 to 63). X2 is not necessarily input 2 (depends where MF 58 is)</p> <p>X(00) = MI1 56 X(01) = MI2 57 X(02) = MI3 58 X(03) = MI4 59 X(04) = MI5 60 X(05) = MI6 61 X(06) = MI7 62 X(07) = MI8 63 (EA terminal in MX2, expansion I/O in RX)</p> | | |

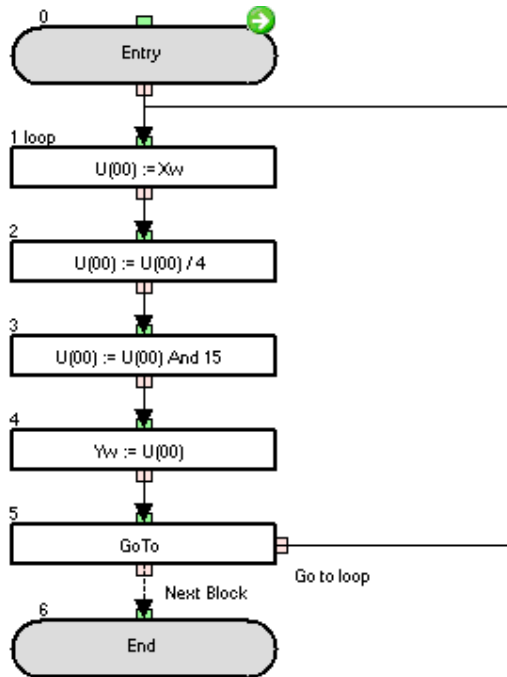
Example



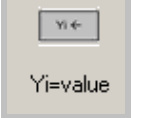
The state of the input terminal X(01) is monitored on the d025 parameter.

| Var = Xw | | |
|---|---|----------------------------------|
| Command | Description | Arguments |
|  | Instruction to access contact inputs by word. Each bit reflects one of the inputs. | • Variable: any variable. |
| Format | | |
| <variable> = Xw | | |
| Note: The inputs have to be assigned to digital multifunction input (by the multifunction 56 to 63) | | |
| Xw = 1 → bit 0 Xw = 2 → bit 1 Xw = 4 → bit 2 Xw = 8 → bit 3 Xw = 16 → bit 4 Xw = 32 → bit 5 Xw = 64 → bit 6 Xw = 128 → bit 7 (only for RX) | | |

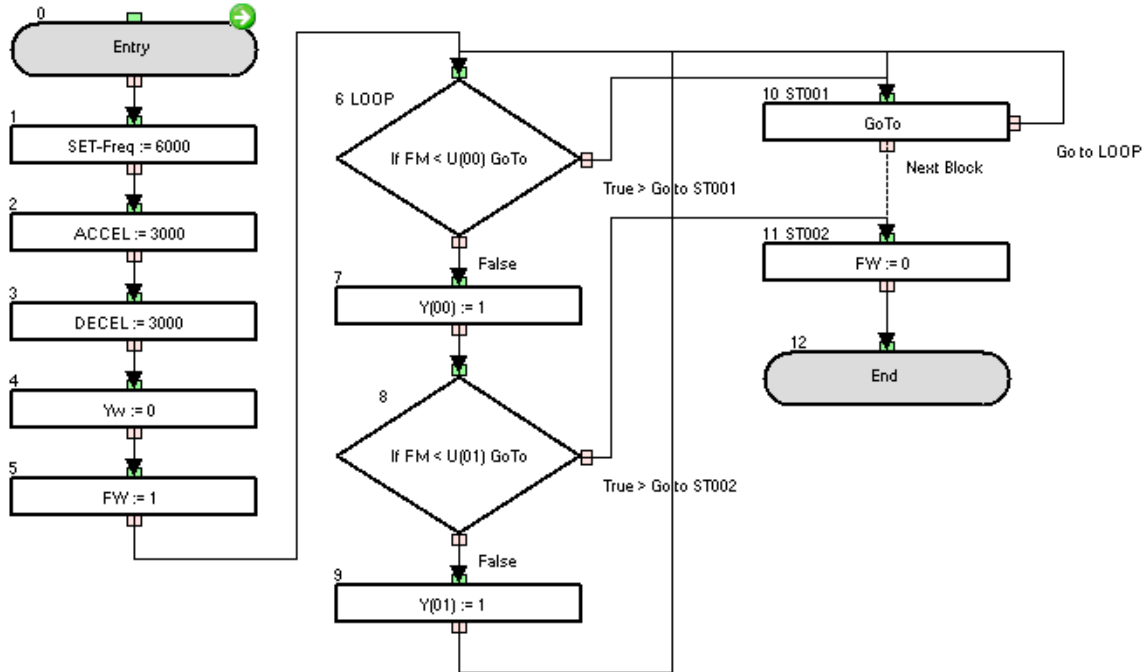
Example



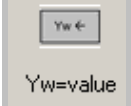
This example acquires the state of the X(02) - X(05) input terminals and outputs it to Y(00) -Y(03) output terminals. To cut X(00) - X(01), the U(00) value is divided by 4. To cut X(06) - X(07), the U(00) value is masked by 15.

| Y(i) = value | | |
|---|--|--|
| Command | Description | Arguments |
|  | Instruction to access digital outputs. | <ul style="list-style-type: none"> • i: Number of the contact output (range 0 to 5) • Value: any variable or constant. |
| Format | | |
| Y(i)=<value> | | |
| <p>Note: The inputs have to be assigned to digital multifunction output (by the multifunction 44 to 49).</p> <p>Y(00) = MO1 44 Y(01) = MO2 45 Y(02) = MO3 46 Y(03) = MO4 47 (with expansion I/O board) Y(04) = MO5 48 (with expansion I/O board) Y(05) = MO6 49 (with expansion I/O board, if enough outputs)</p> | | |

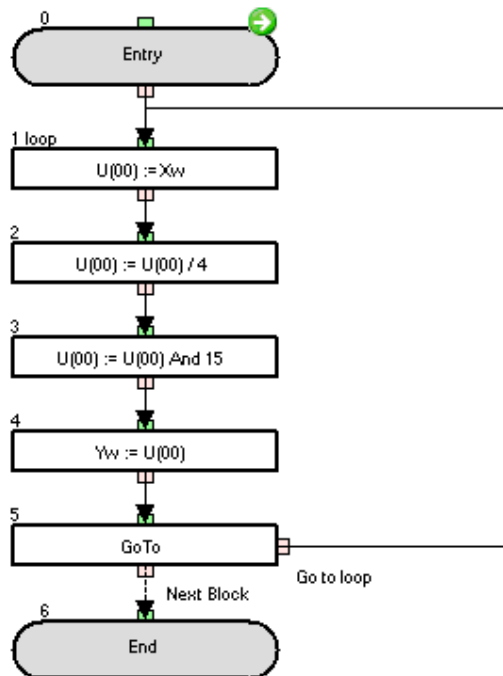
Example:



To test this example, initialize the user variables with the following value: U(00) = 1000, U(01) = 2000, U(02) = 3000. Y(00) – Y(01) are sequentially turned on every 10Hz step of the output frequency.

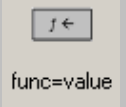
| Yw = value | | |
|---|---|--|
| Command | Description | Arguments |
|  | Instruction to access digital outputs by word. Each bit reflects one of the outputs. | • Value: any variable or constant |
| Format | | |
| Yw = <value> | | |
| Note: The inputs have to be assigned to digital multifunction input (by the multifunction 44 to 49). Yw = 1 → bit 0 Yw = 2 → bit 1 Yw = 4 → bit 2 Yw = 8 → bit 3 (only if expanded I/O board used) Yw = 16 → bit 4 (only if expanded I/O board used) Yw = 32 → bit 5 (only if expanded I/O board used, and enough outputs) | | |

Example:

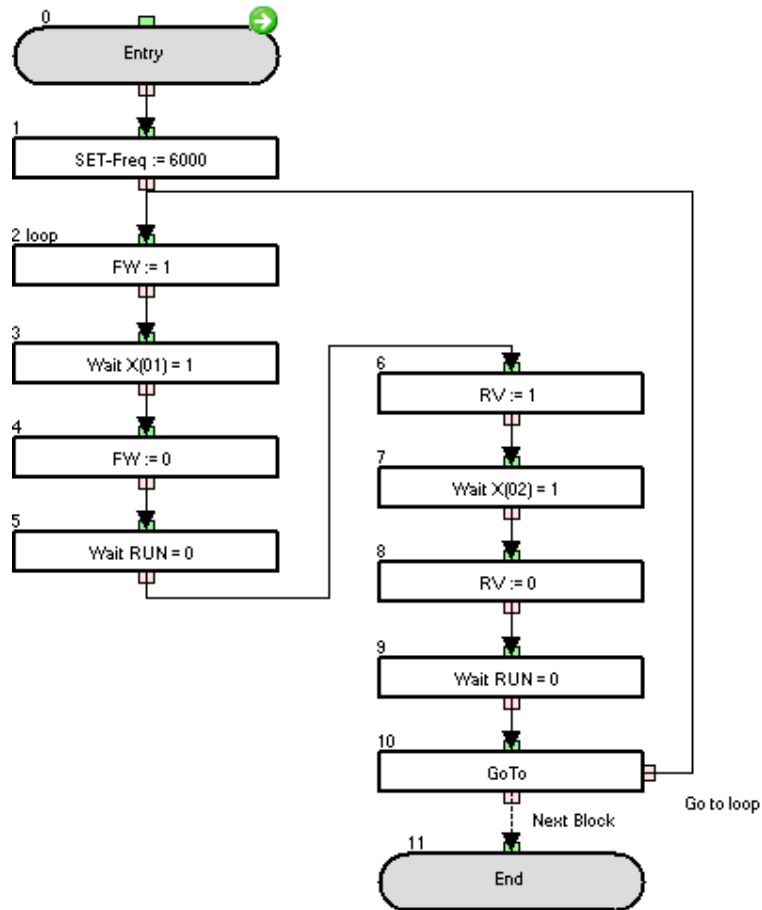


This example acquires the state of the X(02)-X(05) input terminals and outputs it to Y(00)-Y(03) output terminals.

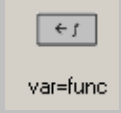
To cut X(00) – X(01), the U(00) value is divided by 4. To cut X(06) – X(07), the U(00) value is masked by 15.

| func = value | | |
|---|---|---|
| Command | Description | Arguments |
|  | Assigns the value of a variable to a command of a terminal input. | <ul style="list-style-type: none"> • Function: any function of input terminal. • Value: any variable or constant. |
| Format | | |
| <function> = <value> | | |

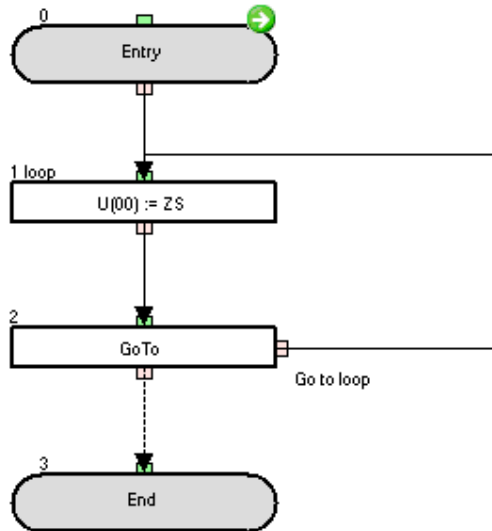
Example



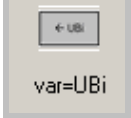
A forward and reverse run at 60Hz is repeated continuously.

| Var = func | | |
|---|---|---|
| Command | Description | Arguments |
|  var=func | A terminal output status is assigned to a variable. | <ul style="list-style-type: none"> • Variable: any variable. • Function: any function of output terminal. |
| Format | | |
| <variable>=<function> | | |

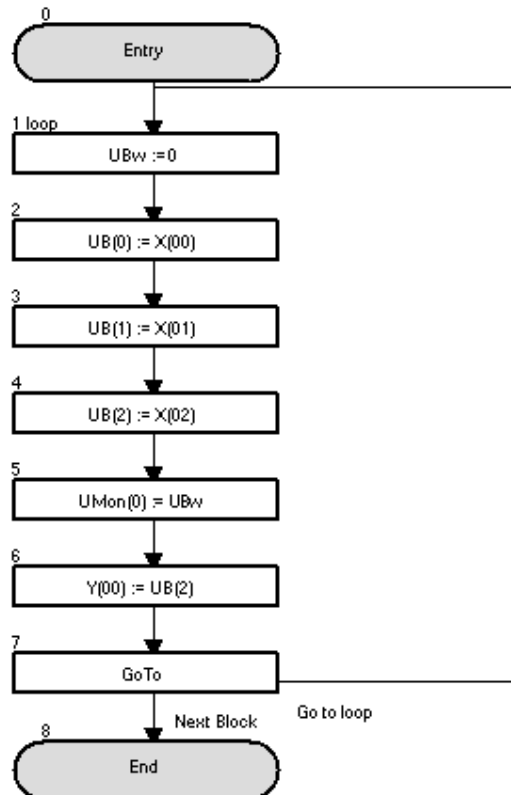
Example




The value of P100 is set to "1" if the ZS (zero speed signal) is on, otherwise is set to "0".

| Var = UB(i) | | |
|---|--|--|
| Command | Description | Arguments |
|  | Assigns the value of an internal user contact to a variable. | <ul style="list-style-type: none"> • Variable: any variable (the value of the variable will be 0 or 1). • i: Number of the user contact (range 0 to 7) |
| Format | | |
| <variable> = UB(i) | | |

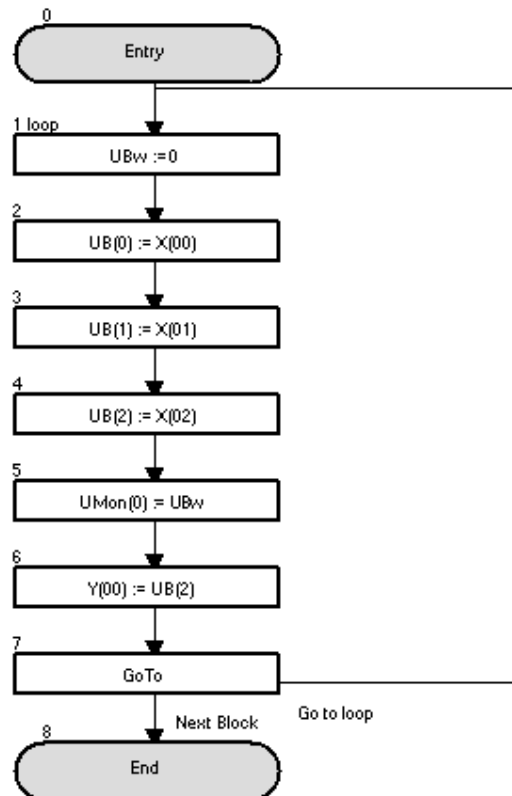
Example




The internal user contacts are cleared on the loop's 1st instruction.
 The status of the X(00) – X(02) input terminals are stored in the UB(0) – UB(2) internal user contacts and monitored on the d025 parameter.
 Finally, the status of the X(02) input terminal is set tot the Y(00) output terminal.

| Var = UBw | | |
|---|--|--|
| Command | Description | Arguments |
|  | Assigns the value of the internal user contact as word (all together) to a word variable. . | <ul style="list-style-type: none"> • Variable: any variable. |
| Format | | |
| <variable> = UBw | | |
| Note: UBw = 1 → bit 0 UBw = 2 → bit 1 UBw = 4 → bit 2 UBw = 8 → bit 3 UBw = 16 → bit 4 UBw = 32 → bit 5 UBw = 64 → bit 6 UBw = 128 → bit 7 | | |

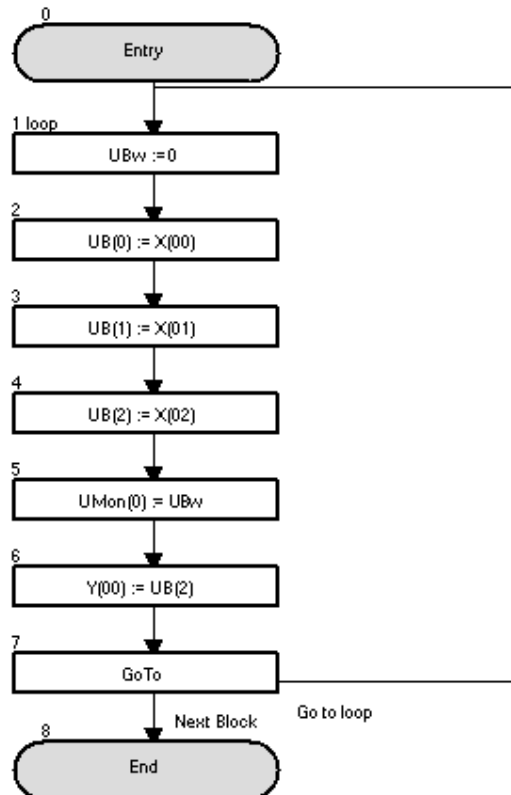
Example



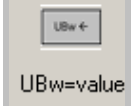
The internal user contacts are cleared on the loop's 1st instruction.
 The status of the X(00) – X(02) input terminals are stored in the UB(0) – UB(2) internal user contacts and monitored on the d025 parameter.
 Finally the status of the X(02) input terminal is set to the Y(00) output terminal.

| UB(i) = value | | |
|---|--|---|
| Command | Description | Arguments |
|  | Assigns a value to an internal user contact control. | <ul style="list-style-type: none"> • i: Number of the user contact (range 0 to 7). • Value: any variable or constant. |
| Format | | |
| UB(i) = <value> | | |

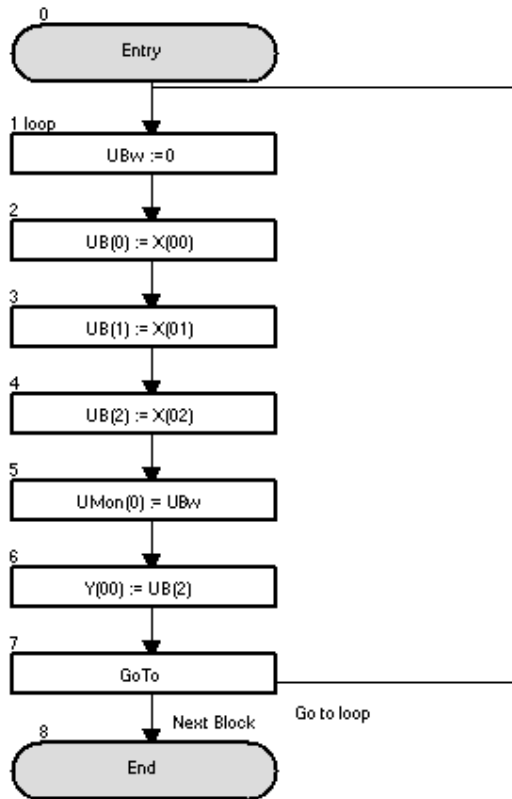
Example



The internal user contacts are cleared on the loop's 1st instruction. The status of the X(00)-X(02) input terminals are stored in the UB(0)-UB(2) internal user contacts and monitored on the d025 parameter. Finally, the status of the X(02) input terminal is set to the Y(00) output terminal.


| UBw = value | | |
|---|--|---|
| Command | Description | Arguments |
|  | Assigns a value to the internal user contact controls. Instruction to access internal user contact by word. | • Value: any variable or constant. |
| Format | | |
| UBw = <value> | | |
| Note: UBw = 1 → bit 0 UBw = 2 → bit 1 UBw = 4 → bit 2 UBw = 8 → bit 3 UBw = 16 → bit 4 UBw = 32 → bit 5 UBw = 64 → bit 6 UBw = 128 → bit 7 | | |

Example

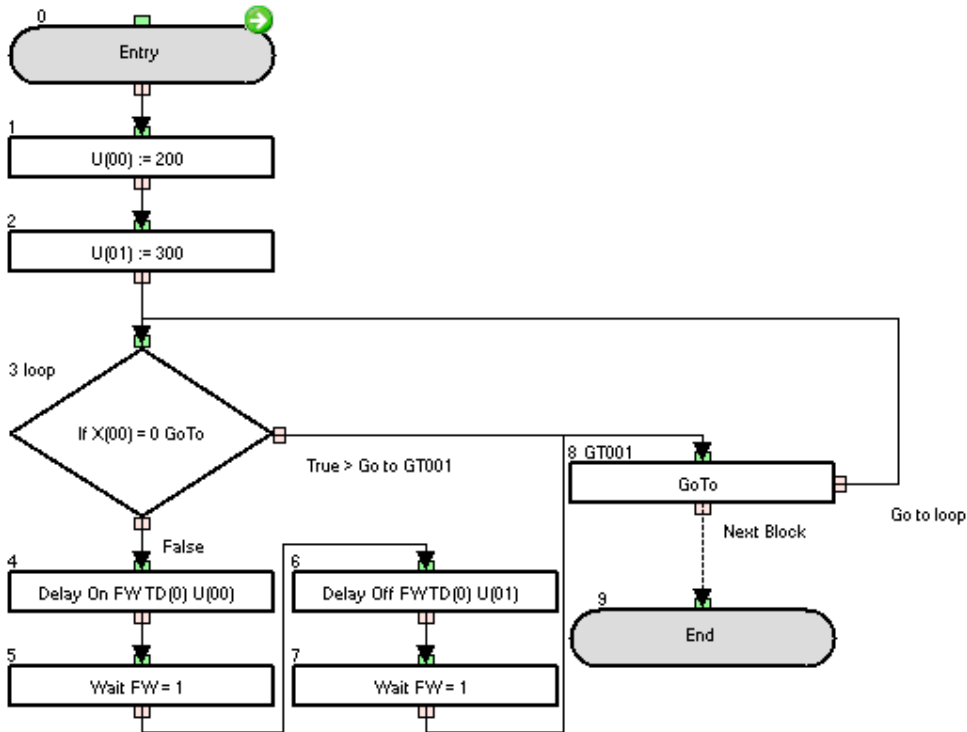


The internal user contacts are cleared on the loop's 1st instruction. The status of the X(00)-X(02) input terminals are stored in the UB(0)-UB(2) internal user contacts and monitored on the d025 parameter. Finally, the status of the X(02) input terminal is set to the Y(00) output terminal.

6-4 Timer Control Commands

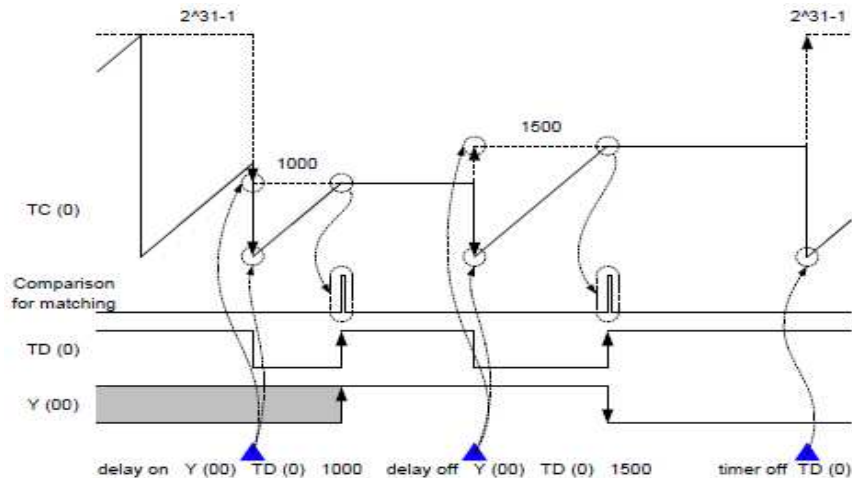
| Delay | | |
|---|---|---|
| Command | Description | Arguments |
|  | This instruction sets the count of the timer in <value 2> and starts the timer counter. When the timer output "TD (K)" is turned on/off, <value 1> is turned on/off. It is important to note, that meantime counting proceeds, the <value 1> remains unchanged from original value. | <ul style="list-style-type: none"> • Value 1: any variable. • Value 2: any variable or constant (time in 10 x ms) • K: number of timer. |
| Format | | |
| Delay on/off <value 1>TD(k)<value 2> | | |


Example



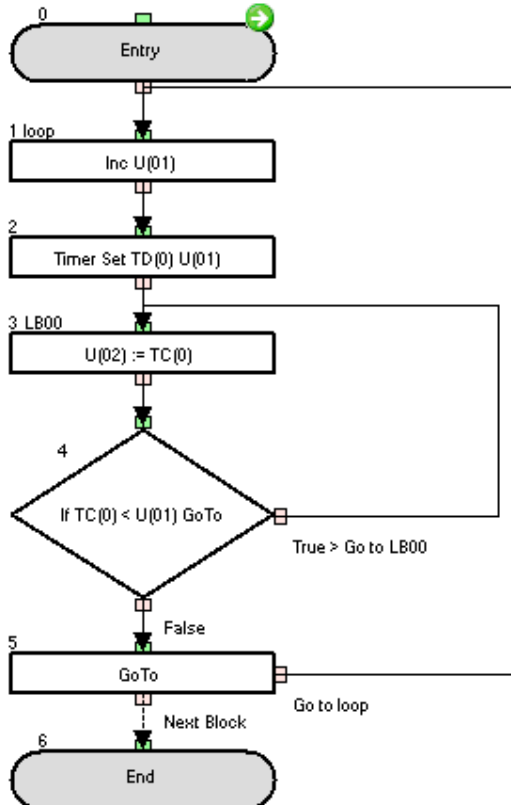
Sample program that activates/deactivates the FW instruction with Delay On/Delay Off instruction.

Timing chart



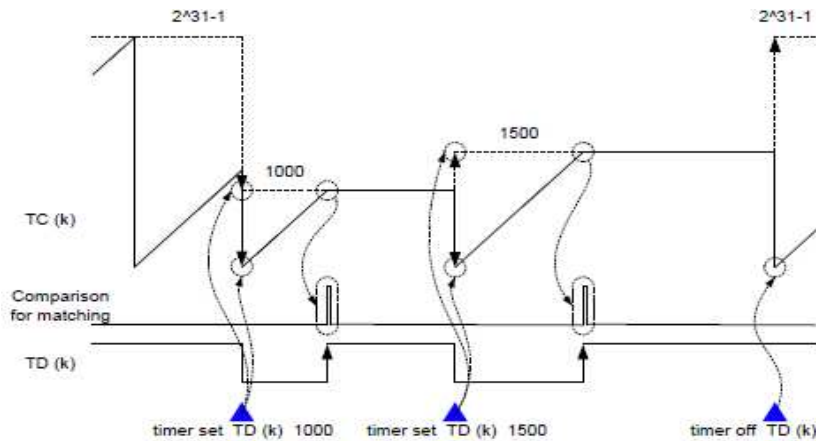
| Timer Set | | |
|---|---|--|
| Command | Description | Arguments |
|  | Sets <value> in the timer and starts the timer counter. The timer starts counting at 0 and increments until <value>. Associated timer contact reflects status ("1" = finish timing) | <ul style="list-style-type: none"> • Value: any variable or constant (time in 10 x ms) • K: number of timer (range 0 to 7) |
| Format | | |
| Timer set TD(k) <value> | | |
| Note: Timer value can be check in variable TC(k). Completion of timer can be checked in variable TD(k) (it becomes "1"). | | |


Example



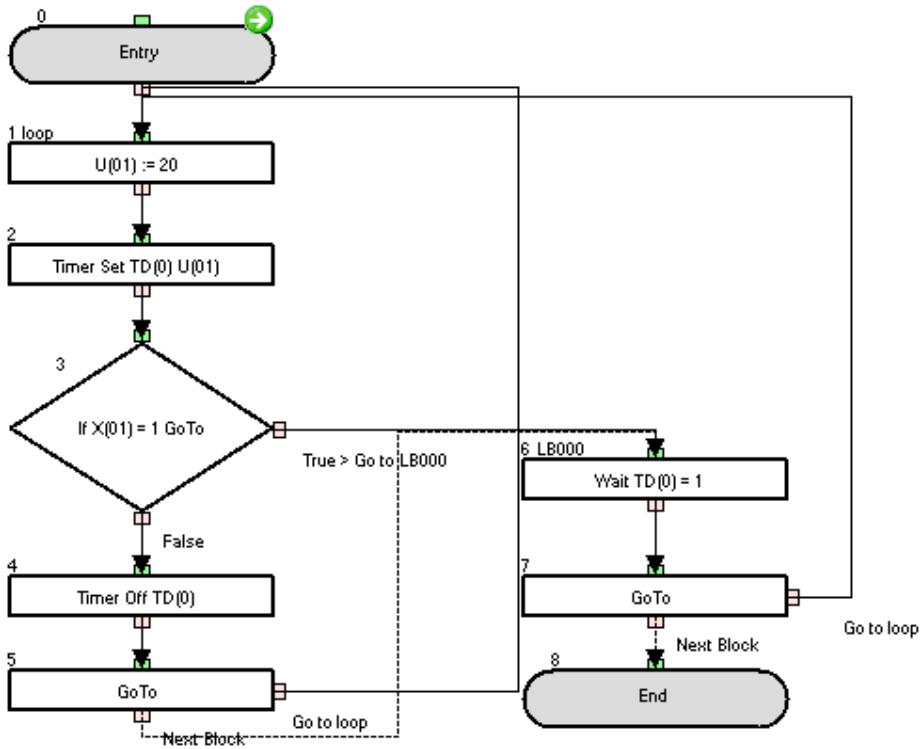
This program will set the timer TD(0) to an increasing value each timer execution. Therefore, because of increasing U(01), it will take longer every cycle to close the loop.

Timing chart



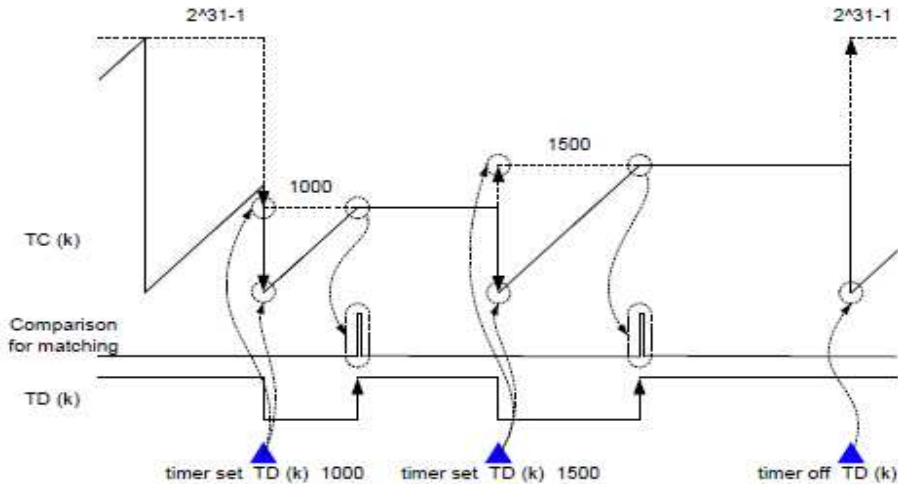
| Timer Off | | |
|---|---|--|
| Command | Description | Arguments |
|  TimerOff | Clears the timer counter (up counter) to zero, and starts the timer counter in free-running timer mode. | <ul style="list-style-type: none"> • k: number of timer (range 0 to 7) |
| Format | | |
| Timer off TD(k) | | |

Example




This example uses a fixed timer execution. But it is cancelled when digital input X(01) is OFF.

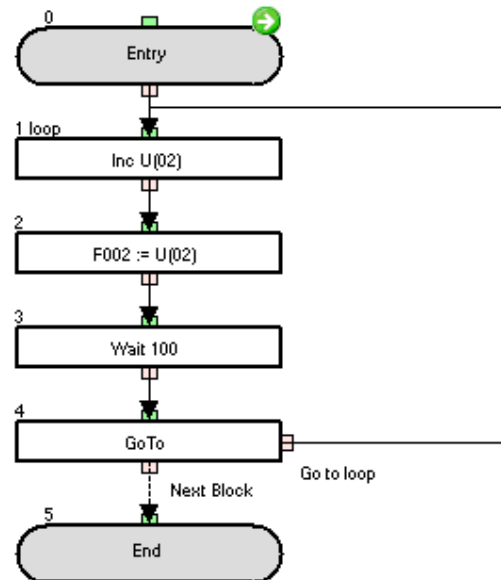
Timing chart




6-5 Parameter Control Commands

| ChgParam | | |
|---|---|--|
| Command | Description | Arguments |
|  ChgParam | Changes the parameter's inverter setting specified by display code to a value. Any inverter parameter can be changed. | <ul style="list-style-type: none"> • Parameter: parameter code (Fxxx, Axxx, bXXX, Cxxx, Hxxx, Pxxx) • Value: any variable or constant. |
| Format | | |
| ChgParam <parameter><value> | | |
| Note: The same rules to parameter writing from operator panel or communications apply: Some parameters can not be written in certain mode of inverter (e.g. some parameters can not be changed during RUN condition). This instruction does not fix the parameter in EEPROM (EepWrt to be used for this purpose) | | |

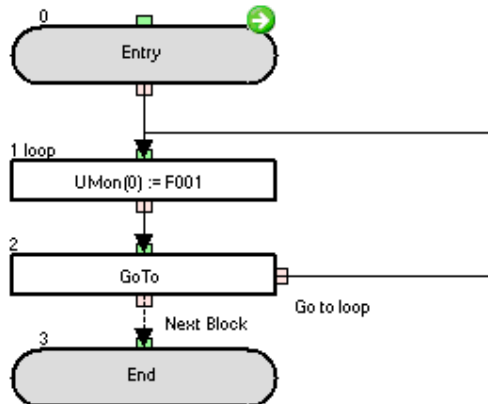
Example




The F002 (acceleration time setting 1) value is increased by 1 every second.

| MonParam | | |
|---|---|---|
| Command | Description | Arguments |
|  MonParam | Assigns the inverter's parameter content specified by display code to a variable. | <ul style="list-style-type: none"> • Parameter: parameter code (Fxxx, Axxx, bxxx, Cxxx, dxxx, Hxxx, Pxxx). • Variable: any variable |
| Format | | |
| MonParam<parameter><variable> | | |

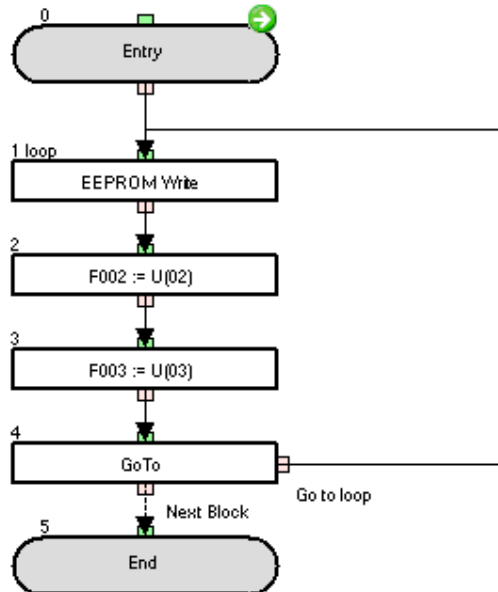
Example




The value of the F001 parameter (output frequency setting) is monitored on the d025 parameter (user parameter monitor).

| EepWrt | | |
|--|--|-----------|
| Command | Description | Arguments |
|  | The command allows write into EEPROM for the next ChgParam immediately following. (if two ChgParam following EepWrt, only for the first will be applied). | --- |
| Format | | |
| EepWrt | | |
| Note: Limitation of EepWrt . | | |
| <p>-If this command is executed in more than one task, ChgParam is executed in the sequence it is detected, but for the second invocation of the command, a waiting time of typically 10 ms will occur before each ChgParam executed. For example, when ChgParam is detected in task 1,2 and 3 at the same time, and the one in task 1 is executed at first, 10 ms for task 2 and 20 ms for task 3 are two waited. But when Eepwrt is not executed, ChgParam don't need the waiting time.</p> | | |

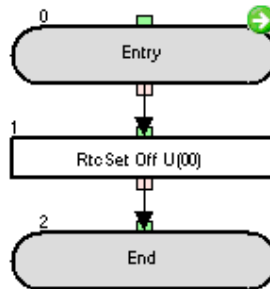
Example: (only F002 is stored in EEPROM)



On executing the program, only F002 parameter is saved permanently from U(02). After power off and on again, F003 will have the old value. The initial values of the U(02) and U(03) variables can be set on the program variables list or the P102, P103 parameters.

| RtcSet | | |
|--|--|---|
| Command | Description | Arguments |
|  | <p>This statement sets 6 bytes data of time to a variable. The 6 bytes data of time means year, month, day, a day of week, hour and minute.</p> <p>The variable value in hexadecimal corresponds to the year, month, day, day of a week, hour and minute (in decimal).</p> <p>RtcSet on: updates the 6 bytes data continuously.</p> <p>RtcSet off: updates the 6 bytes data only once.</p> | <ul style="list-style-type: none"> • User variable: any user or internal user variable (U(xx) or UL(xx)). |
| Format | | |
| RtcSet on/off <user variable> | | |
| <p>Note:</p> <ul style="list-style-type: none"> • RtcSet on U(<k>): It will set U(<k>) with 2 bytes for year and 2 bytes for month, U(<k+1>) with 2 bytes for Month's day and 2 bytes for week's day(00 for Sunday, 06 for Saturday), and U(<k+2>) with 2 bytes for hour and 2 bytes for minutes. • RtcSet on UL(<k>): It will set UL(<k>) with 2 bytes for year, 2 bytes for month, 2 bytes for month's day and 2 bytes for week's day (00 for Sunday, 06 for Saturday), and UL(<k+1>) with 2 bytes for hour, 2 bytes for minutes and 4 bytes of padding(0000). • If the watch LCD operator is not attached, RtcSet instruction sets 000000000000h | | |

Example





After executing the program (with the watch LCD operator attached), the hexadecimal value of the first 2 bytes of U(00) will correspond with the current year and the hexadecimal value of the last 2 bytes of U(00) will correspond to the current month.


I.e. if the example program runs on July 5th (Monday) of 2010 at 02:29 P.M., then U(00), U(01) and U(02) will display the following values:


| Parameter... | ...display in decimal format... | Which converted to hexadecimal format results in... | ...which means |
|--------------|---------------------------------|---|---|
| U(00) | 4103 | 1007 | '10' for 2010 '07' for July |
| U(01) | 1281 | 0501 | '05 for 5 th day of month '01' for Monday |
| U(02) | 5161 | 1429 | '14' for 2 p.m. '29' for 29 minutes |

6-6 Inverter Control Commands

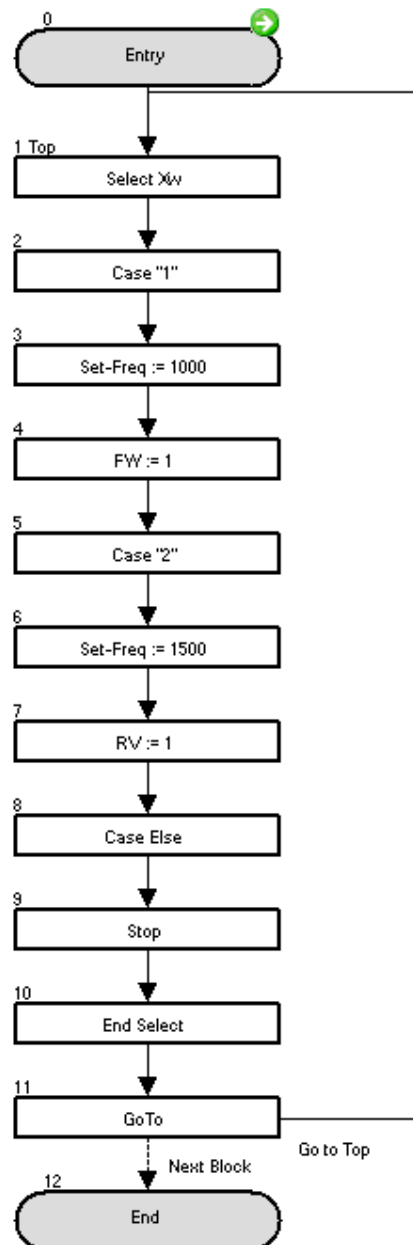
| Run FW | | |
|---|---|-----------|
| Command | Description | Arguments |
|  | Makes the inverter run the motor in forward direction (starts the inverter output). This command is a shortcut of the func = value command. | --- |
| Format | | |
| FW = 1 | | |
| Note: The instruction is available since CX-Drive v2.10. | | |

| Run RV | | |
|---|---|-----------|
| Command | Description | Arguments |
|  | Makes the inverter run the motor in reverse direction (starts the inverter output). This command is a shortcut of the func = value command. | --- |
| Format | | |
| RV = 1 | | |
| Note: This instruction is available since CX-Drive v2.10. | | |


| Stop | | |
|--|--|-----------|
| Command | Description | Arguments |
|  | Makes the inverter decelerate and stop the motor (stop the inverter output). | --- |
| Format | | |
| Stop | | |

| Set Freq | | |
|---|--|---|
| Command | Description | Arguments |
|  | It sets the frequency of the inverter. This command is a shortcut of the '=' command. Units: 0.01Hz. | <ul style="list-style-type: none"> • Value: any variable or constant (range from 0 to 40000). |
| Format | | |
| Set-Freq = <value> | | |
| Note: This instruction is available since CX-Drive v2.10. | | |

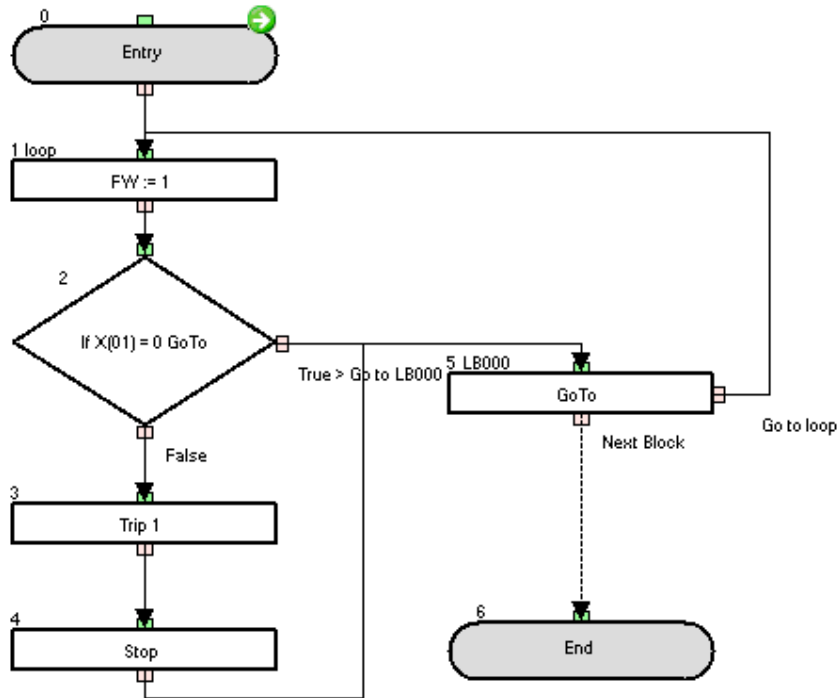
Example



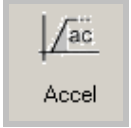
This program will run the motor in forward direction at 10Hz if general input contact Xw is 1. If general input contact Xw is 2, it will run in reverse direction at 15Hz. For other values the motor will stop.


| Trip | | |
|---|---------------------------------------|--|
| Command | Description | Arguments |
|  Trip | This instruction makes inverter trip. | <ul style="list-style-type: none"> • Value: any variable or constant (range 0 to 9). |
| Format | | |
| Trip<value> | | |

Example

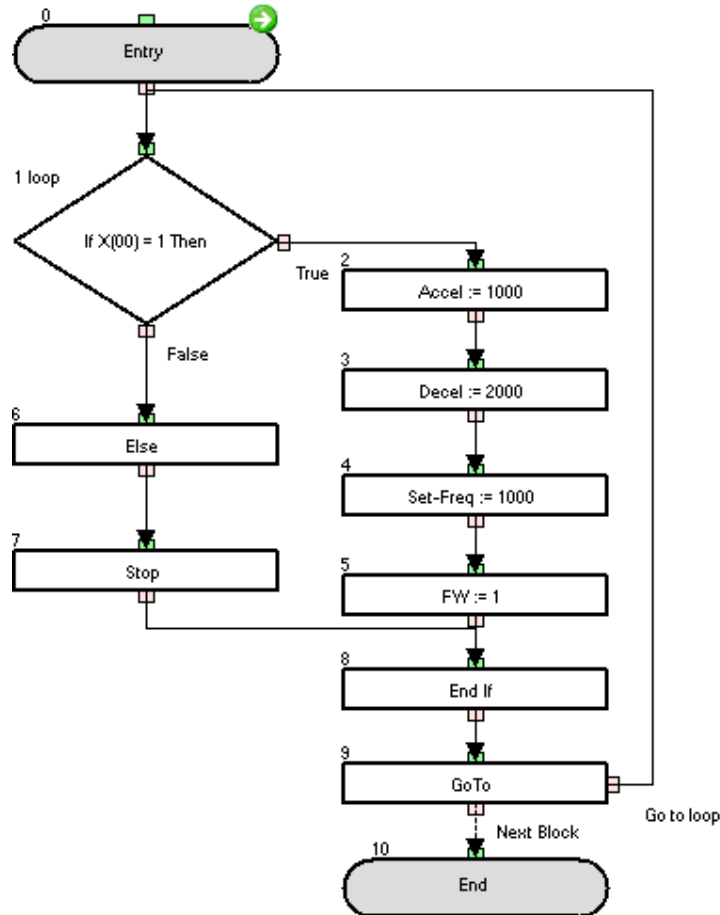


This sample program will throw a user trip on the inverter when digital input X(01) is set to ON.

| Accel | | |
|--|---|--|
| Command | Description | Arguments |
|  | It sets the acceleration time of the inverter. This command is a shortcut of the '=' command. Units: 10 ms. | <ul style="list-style-type: none"> • Value: any variable or constant (range from 1 to 360000). |
| Format | | |
| Accel = <value> | | |
| Note: The P031 parameter must be set to value 3 (Drive Programming) for the command has effect. | | |

| Decel | | |
|---|---|--|
| Command | Description | Arguments |
|  | It sets the deceleration time of the inverter. This command is a shortcut of the '=' command. Units: 10ms | <ul style="list-style-type: none"> • Value: any variable or constant (range from 1 to 360000). |
| Format | | |
| Decel = <value> | | |
| Note: The parameter P031 must be set to 3(Drive Programming) for the command has effect. | | |

Example



This sample program will set the Acceleration to 10 seconds and deceleration to 20 seconds if digital input X(00) is set to ON.

7- Drive Programming specific trips and Troubleshooting

The table below shows how to handle the specific errors to Drive Programming function. For details on other errors in the inverter, refer to the inverter instruction manual.

| Factor code | Error (causing inverter trip) | Possible cause | Checking method | Corrective action |
|-------------|-------------------------------|--|---|---|
| E43 | Invalid instruction | The PRG terminal was turned on without a program downloaded to the inverter. | By uploading the program, you can check if really a program is in the inverter or not. | Recreate the program, and then download it to the inverter |
| E44 | Nesting count error | Subroutines are nested in more than eight layers. For-Next loop statements are nested in more than eight layers. If statements are nested in more than eight layers. | Read the program to check the number of nesting layers (some times difficult to recognize)... | Correct the program so that the number of layers will be eight or less. |
| E45 | Instruction error 1 | The jump destination of a GoTo instruction is a next instruction to end a for or other loop. | Check whether each GoTo instruction jumps to an instruction that ends a loop. | Correct the jump destinations of GoTo instructions. As general recommendation, never jump a Goto out of the current level it is. |
| | | The variable "U(ii)" referenced via another variable is not found. | Check the numerical value specified in "U(ii)". | Correct the value of variable "U(ii)" or limit the range of values of variable "U(ii)". |
| | | An arithmetic instruction caused: -Overflow, -underflow, or -division by zero | Check the program for the instruction causing overflow, underflow, or division by zero (not in early MX2 firmware). | Correct the program so that no arithmetic instruction causes overflow, underflow, or division by zero. |
| | | A ChgParam instruction caused: - reference to a non existing parameter. -writing of a value out of the setting range -change of a parameter value (during inverter operation) that cannot be updated during inverter operation, or Change of a parameter value of which updating is restricted by software lock (when software lock is enabled). | -Check the parameters and the values to be written. -If the error has occurred during inverter operation, check whether the parameter in question can be updated during inverter operation. -Check the setting of software lock selection (b031). | -Correct the parameters or the values to be written to parameters so that they will be within the setting range. -Disable software lock. -If the parameter to be updated is the one that cannot be updated during inverter operation, change the setting of software lock selection (b031) to "10" to switch to the mode enabling parameter updating during inverter operation. |
| E50 to E59 | User trip 0 to 9 | These trips are generated from the user application. The cause is determined by the Drive Programming logic | Check with the drive program documentation to recognize the trip conditions | Check the drive program documentation to recognize countermeasures |

8- Drive Programming Parameters – General Precautions

8-1 Parameters list affected by setting order

| Parameter | Description |
|-----------|--|
| A003 | Base frequency setting |
| A004 | Maximum frequency setting |
| A203 | Base frequency setting, 2 nd motor |
| A204 | Maximum frequency setting, 2 nd motor |
| B015 | Free setting, electronic thermal frequency (1) |
| B017 | Free setting, electronic thermal frequency (2) |
| B019 | Free setting, electronic thermal frequency (3) |
| B049 | Dual Rating Selection |
| B050 | Controlled deceleration on power loss |
| B051 | DC bus voltage trigger level of control deceleration |
| B052 | Over-voltage threshold of control deceleration |
| B060 | Maximum-limit level of window comparators O |
| B061 | Minimum-limit level of window comparators O |
| B062 | Hysteresis width of windows comparators O |
| B063 | Maximum-limit level of window comparators OI |
| B064 | Minimum-limit level of window comparators OI |
| B065 | Hysteresis width of window comparator (OI) |
| B079 | Watt-hour display gain setting |
| B082 | Start frequency adjustment |
| B100 | Free setting V/f freq. (1) |
| B102 | Free setting V/f freq. (2) |
| B104 | Free setting V/f freq. (3) |
| B106 | Free setting V/f freq. (4) |
| B108 | Free setting V/f freq. (5) |
| B110 | Free setting V/f freq. (6) |
| B112 | Free setting V/f freq. (7) |
| P070 | Low-speed zero-return frequency |

8-2 Parameters list affected by Rated Current (%)

| Parameter | Description |
|-----------|--|
| B012 | Level of electronic thermal setting |
| B016 | Free setting, electronic thermal current (1) |
| B018 | Free setting, electronic thermal current (2) |
| B020 | Free setting, electronic thermal current (3) |
| B022 | Overload restriction level setting |
| B025 | Overload restriction level 2 setting |
| B028 | Current level of active freq. matching restart setting |
| B126 | Brake release current setting |
| B212 | Level of electronic thermal setting, 2 nd motor |
| B222 | Overload restriction operation mode, 2 nd motor |
| C030 | Digital current monitor reference value |
| C039 | Low load detection level |
| C041 | Overload level setting |
| C111 | Overload setting (2) |
| C241 | Overload level setting, 2 nd motor |

8-3 Parameters list affected by PID enabled/disabled

| Parameter | Description |
|-----------|--|
| A011 | Pot./O-L input active range start frequency |
| A012 | Pot./O-L input active range end frequency |
| A020 | Multi-speed 0 setting |
| A021 | Multi-speed 1 setting |
| A022 | Multi-speed 2 setting |
| A023 | Multi-speed 3 setting |
| A024 | Multi-speed 4 setting |
| A025 | Multi-speed 5 setting |
| A026 | Multi-speed 6 setting |
| A027 | Multi-speed 7 setting |
| A028 | Multi-speed 8 setting |
| A029 | Multi-speed 9 setting |
| A030 | Multi-speed 10 setting |
| A031 | Multi-speed 11 setting |
| A032 | Multi-speed 12 setting |
| A033 | Multi-speed 13 setting |
| A034 | Multi-speed 14 setting |
| A035 | Multi-speed 15 setting |
| A101 | [OI] input active Range start frequency |
| A102 | [OI] input active Range end frequency |
| A145 | ADD frequency |
| A220 | Multi-speed 0 setting, 2 nd motor |
| F001 | Output frequency setting |

These parameters are affected by A071 / A075.

Terms and Conditions of Sale

1. **Offer; Acceptance.** These terms and conditions (these "Terms") are deemed part of all quotes, agreements, purchase orders, acknowledgments, price lists, catalogs, manuals, brochures and other documents, whether electronic or in writing, relating to the sale of products or services (collectively, the "Products") by Omron Electronics LLC and its subsidiary companies ("Omron"). Omron objects to any terms or conditions proposed in Buyer's purchase order or other documents which are inconsistent with, or in addition to, these Terms.
2. **Prices; Payment Terms.** All prices stated are current, subject to change without notice by Omron. Omron reserves the right to increase or decrease prices on any unshipped portions of outstanding orders. Payments for Products are due net 30 days unless otherwise stated in the invoice.
3. **Discounts.** Cash discounts, if any, will apply only on the net amount of invoices sent to Buyer after deducting transportation charges, taxes and duties, and will be allowed only if (i) the invoice is paid according to Omron's payment terms and (ii) Buyer has no past due amounts.
4. **Interest.** Omron, at its option, may charge Buyer 1-1/2% interest per month or the maximum legal rate, whichever is less, on any balance not paid within the stated terms.
5. **Orders.** Omron will accept no order less than \$200 net billing.
6. **Governmental Approvals.** Buyer shall be responsible for, and shall bear all costs involved in, obtaining any government approvals required for the importation or sale of the Products.
7. **Taxes.** All taxes, duties and other governmental charges (other than general real property and income taxes), including any interest or penalties thereon, imposed directly or indirectly on Omron or required to be collected directly or indirectly by Omron for the manufacture, production, sale, delivery, importation, consumption or use of the Products sold hereunder (including customs duties and sales, excise, use, turnover and license taxes) shall be charged to and remitted by Buyer to Omron.
8. **Financial.** If the financial position of Buyer at any time becomes unsatisfactory to Omron, Omron reserves the right to stop shipments or require satisfactory security or payment in advance. If Buyer fails to make payment or otherwise comply with these Terms or any related agreement, Omron may (without liability and in addition to other remedies) cancel any unshipped portion of Products sold hereunder and stop any Products in transit until Buyer pays all amounts, including amounts payable hereunder, whether or not then due, which are owing to it by Buyer. Buyer shall in any event remain liable for all unpaid accounts.
9. **Cancellation; Etc.** Orders are not subject to rescheduling or cancellation unless Buyer indemnifies Omron against all related costs or expenses.
10. **Force Majeure.** Omron shall not be liable for any delay or failure in delivery resulting from causes beyond its control, including earthquakes, fires, floods, strikes or other labor disputes, shortage of labor or materials, accidents to machinery, acts of sabotage, riots, delay in or lack of transportation or the requirements of any government authority.
11. **Shipping; Delivery.** Unless otherwise expressly agreed in writing by Omron:
 - a. Shipments shall be by a carrier selected by Omron; Omron will not drop ship except in "break down" situations.
 - b. Such carrier shall act as the agent of Buyer and delivery to such carrier shall constitute delivery to Buyer;
 - c. All sales and shipments of Products shall be FOB shipping point (unless otherwise stated in writing by Omron), at which point title and risk of loss shall pass from Omron to Buyer; provided that Omron shall retain a security interest in the Products until the full purchase price is paid;
 - d. Delivery and shipping dates are estimates only; and
 - e. Omron will package Products as it deems proper for protection against normal handling and extra charges apply to special conditions.
12. **Claims.** Any claim by Buyer against Omron for shortage or damage to the Products occurring before delivery to the carrier must be presented in writing to Omron within 30 days of receipt of shipment and include the original transportation bill signed by the carrier noting that the carrier received the Products from Omron in the condition claimed.
13. **Warranties.** (a) **Exclusive Warranty.** Omron's exclusive warranty is that the Products will be free from defects in materials and workmanship for a period of twelve months from the date of sale by Omron (or such other period expressed in writing by Omron). Omron disclaims all other warranties, express or implied. (b) **Limitations.** OMRON MAKES NO WARRANTY OR REPRESENTATION, EXPRESS OR IMPLIED, ABOUT NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OF THE PRODUCTS. BUYER ACKNOWLEDGES THAT IT ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE. Omron further disclaims all warranties and responsibility of any type for claims or expenses based on infringement by the Products or otherwise of any intellectual property right. (c) **Buyer Remedy.** Omron's sole obligation hereunder shall be, at Omron's election, to (i) replace (in the form originally shipped with Buyer responsible for labor charges for removal or replacement thereof) the non-complying Product, (ii) repair the non-complying Product, or (iii) repay or credit Buyer an amount equal to the purchase price of the non-complying Product; provided that in no event shall Omron be responsible for warranty, repair, indemnity or any other claims or expenses regarding the Products unless Omron's analysis confirms that the Products were properly handled, stored, installed and maintained and not subject to contamination, abuse, misuse or inappropriate modification. Return of any Products by Buyer must be approved in writing by Omron before shipment. Omron Companies shall not be liable for the suitability or unsuitability or the results from the use of Products in combination with any electrical or electronic components, circuits, system assemblies or any other materials or substances or environments. Any advice, recommendations or information given orally or in writing, are not to be construed as an amendment or addition to the above warranty. See <http://www.omron247.com> or contact your Omron representative for published information.
14. **Limitation on Liability; Etc.** OMRON COMPANIES SHALL NOT BE LIABLE FOR SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR PRODUCTION OR COMMERCIAL LOSS IN ANY WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED IN CONTRACT, WARRANTY, NEGLIGENCE OR STRICT LIABILITY. Further, in no event shall liability of Omron Companies exceed the individual price of the Product on which liability is asserted.
15. **Indemnities.** Buyer shall indemnify and hold harmless Omron Companies and their employees from and against all liabilities, losses, claims, costs and expenses (including attorney's fees and expenses) related to any claim, investigation, litigation or proceeding (whether or not Omron is a party) which arises or is alleged to arise from Buyer's acts or omissions under these Terms or in any way with respect to the Products. Without limiting the foregoing, Buyer (at its own expense) shall indemnify and hold harmless Omron and defend or settle any action brought against such Companies to the extent based on a claim that any Product made to Buyer specifications infringed intellectual property rights of another party.
16. **Property; Confidentiality.** Any intellectual property in the Products is the exclusive property of Omron Companies and Buyer shall not attempt to duplicate it in any way without the written permission of Omron. Notwithstanding any charges to Buyer for engineering or tooling, all engineering and tooling shall remain the exclusive property of Omron. All information and materials supplied by Omron to Buyer relating to the Products are confidential and proprietary, and Buyer shall limit distribution thereof to its trusted employees and strictly prevent disclosure to any third party.
17. **Export Controls.** Buyer shall comply with all applicable laws, regulations and licenses regarding (i) export of products or information; (ii) sale of products to "forbidden" or other proscribed persons; and (iii) disclosure to non-citizens of regulated technology or information.
18. **Miscellaneous.** (a) **Waiver.** No failure or delay by Omron in exercising any right and no course of dealing between Buyer and Omron shall operate as a waiver of rights by Omron. (b) **Assignment.** Buyer may not assign its rights hereunder without Omron's written consent. (c) **Law.** These Terms are governed by the law of the jurisdiction of the home office of the Omron company from which Buyer is purchasing the Products (without regard to conflict of law principles). (d) **Amendment.** These Terms constitute the entire agreement between Buyer and Omron relating to the Products, and no provision may be changed or waived unless in writing signed by the parties. (e) **Severability.** If any provision hereof is rendered ineffective or invalid, such provision shall not invalidate any other provision. (f) **Setoff.** Buyer shall have no right to set off any amounts against the amount owing in respect of this invoice. (g) **Definitions.** As used herein, "including" means "including without limitation"; and "Omron Companies" (or similar words) mean Omron Corporation and any direct or indirect subsidiary or affiliate thereof.

Certain Precautions on Specifications and Use

1. **Suitability of Use.** Omron Companies shall not be responsible for conformity with any standards, codes or regulations which apply to the combination of the Product in the Buyer's application or use of the Product. At Buyer's request, Omron will provide applicable third party certification documents identifying ratings and limitations of use which apply to the Product. This information by itself is not sufficient for a complete determination of the suitability of the Product in combination with the end product, machine, system, or other application or use. Buyer shall be solely responsible for determining appropriateness of the particular Product with respect to Buyer's application, product or system. Buyer shall take application responsibility in all cases but the following is a non-exhaustive list of applications for which particular attention must be given:
 - (i) Outdoor use, uses involving potential chemical contamination or electrical interference, or conditions or uses not described in this document.
 - (ii) Use in consumer products or any use in significant quantities.
 - (iii) Energy control systems, combustion systems, railroad systems, aviation systems, medical equipment, amusement machines, vehicles, safety equipment, and installations subject to separate industry or government regulations.
 - (iv) Systems, machines and equipment that could present a risk to life or property. Please know and observe all prohibitions of use applicable to this Product.
 NEVER USE THE PRODUCT FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY OR IN LARGE QUANTITIES WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESIGNED TO ADDRESS THE RISKS, AND THAT THE OMRON'S PRODUCT IS PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.
2. **Programmable Products.** Omron Companies shall not be responsible for the user's programming of a programmable Product, or any consequence thereof.
3. **Performance Data.** Data presented in Omron Company websites, catalogs and other materials is provided as a guide for the user in determining suitability and does not constitute a warranty. It may represent the result of Omron's test conditions, and the user must correlate it to actual application requirements. Actual performance is subject to the Omron's Warranty and Limitations of Liability.
4. **Change in Specifications.** Product specifications and accessories may be changed at any time based on improvements and other reasons. It is our practice to change part numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the Product may be changed without any notice. When in doubt, special part numbers may be assigned to fix or establish key specifications for your application. Please consult with your Omron's representative at any time to confirm actual specifications of purchased Product.
5. **Errors and Omissions.** Information presented by Omron Companies has been checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical or proofreading errors or omissions.



OMRON ELECTRONICS LLC • THE AMERICAS HEADQUARTERS • Schaumburg, IL USA • 847.843.7900 • 800.556.6766 • www.omron247.com

OMRON CANADA, INC. • HEAD OFFICE

Toronto, ON, Canada • 416.286.6465 • 866.986.6766
www.omron247.com

OMRON ELETRÔNICA DO BRASIL LTDA • HEAD OFFICE

São Paulo, SP, Brasil • 55.11.2101.6300 • www.omron.com.br

OMRON ELECTRONICS MEXICO SA DE CV • HEAD OFFICE

Apodaca, N.L. • 52.811.156.99.10 • 001.800.556.6766 • mela@omron.com

OMRON ARGENTINA • SALES OFFICE

Cono Sur • 54.11.4783.5300

OMRON CHILE • SALES OFFICE

Santiago • 56.9.9917.3920

OTHER OMRON LATIN AMERICA SALES

54.11.4783.5300

OMRON EUROPE B.V. Wegalaan 67-69, NL-2132 JD, Hoofddorp, The Netherlands. Tel: +31 (0) 23 568 13 00 Fax: +31 (0) 23 568 13 88 www.industrial.omron.eu