MultiStop Plus

OPERATOR MANUAL

D1005 Version 1.74
\\Development\c\Products\Multistop***MultiStop** **plus*** Manuals\multistop plus manual v1.74.doc

Goodwin Electronics Ltd.
3 Bassendale Road
Croft Business Park
Bromborough
Wirral
CH62 3QL United Kingdom

Tel: +44 (0) 151 33 44 555
FAX: +44 (0) 151 334 1616

Documentation

This product is available with two manuals;

MultiStop ***Plus*** Technical Specification Manual
MultiStop ***Plus*** Operators Manual.

This is the operator manual which has been written for an operator to refer to when using the ***MultiStop*** ***Plus*** controller in day - to - day operation.

Please note

The contents of this manual are designed to give the reader an understanding of how the ***MultiStop*** ***Plus*** operates and assumes that he is familiar with the machine onto which the unit is connected. The particulars of the product, and its use are given by Goodwin Electronics Ltd in good faith. However, it is acknowledged that there may be errors and omissions in this document. We shall not be liable for the loss or damage whatsoever arising from the use of any information in, or any omissions from, this document.

CONTENTS

This page intentionall left blank

INTRODUCTION

The ***MultiStop*** ***Plus*** controller is designed to control up to two axes of movement. It is a fully self contained unit incorporating,

- a clear LCD display
- the ability to control of the movement of one or two axes
- digital inputs and digital and analogue outputs
- watchdog output
- RS 485 serial communications link

There are two models one has one axis of motion the other has two. Where this manual refers to two axis the axis for the second axis will not be applicable on a single axis unit.

After switching the power on the unit defaults to the AUTO mode window. This is describe in section 2.

The controller requires a program to be resident in the controller before any automatic cycle can be carried out. However the datumming cycle, and jogging of the axes can always be carried out.

It is assumed that the ***MultiStop*** ***Plus*** has been configured correctly for the machine on which it is to be used.

At some time it will be necessary for an engineer to set the parameters in the Configuration mode, see the **Multistop** ***Plus*** Technical Specification manual for further details.

Every **Multistop** ***Plus*** unit also has an RS485 communication link that allows it to become a slave of a master controller, (a PC or another **Multistop** ***Plus*** unit).

Program data and configuration data are kept in battery backed ram. The battery normally has a ten year life and as its voltage is continuously monitored by the controller. When it requires changing a message will be displayed in the status bar window. The procedure necessary to replace the battery is given in the Multistop Technical Manual.

If it is necessary to disconnect any of the cabling attached to the **Multistop** ***Plus*** controller make sure that the strain relief attachments are used to remove the Klippon 8 way sockets and minimum force is applied to the controller. When reconnecting these sockets ensure that the socket is firmly ‘home’.

FRONT PANEL AND DISPLAY LAYOUT

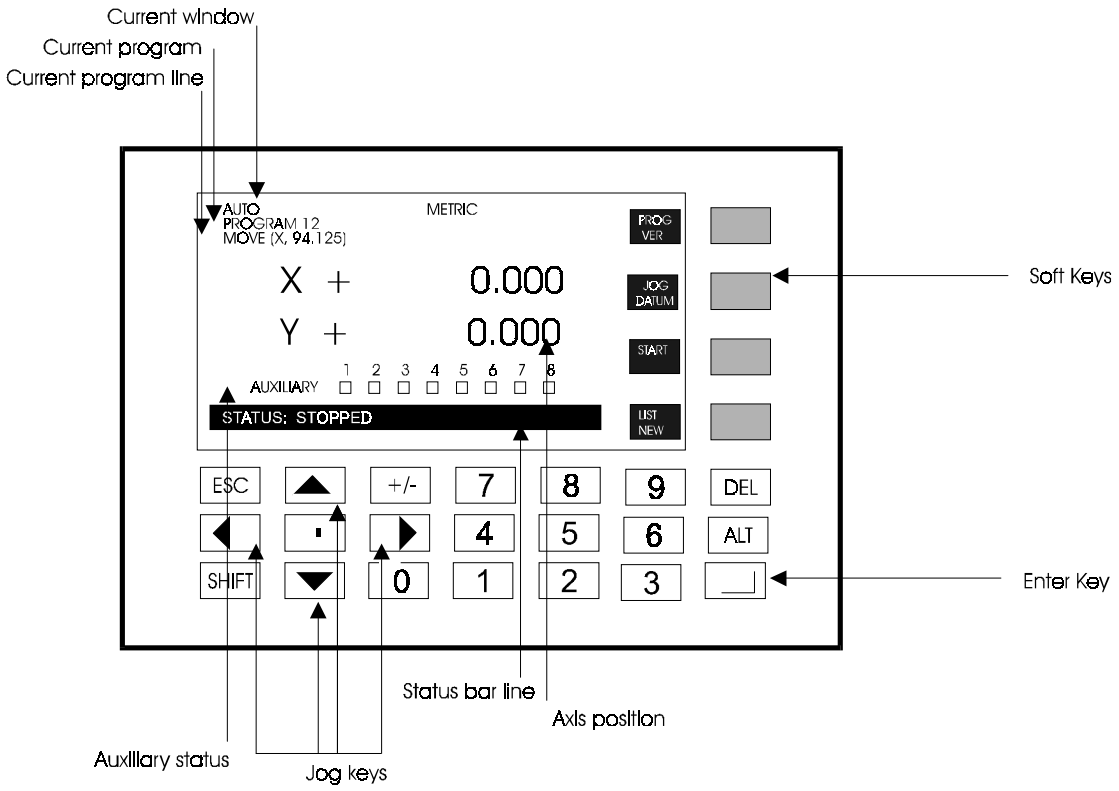


Figure 1

MODES OF OPERATION

In all modes of operation a window will be displayed and the information displayed will consist of,

- four Soft keys whose function will be indicated
- a status bar window in which the current most important message will be displayed
- any other information that is relevant
- access to some modes are password protected. His feature can be canceled by setting the appropriate details in the password parameter page, (see Multistop Technical Specification)

All other keys are divided as follows

- numeric keys used for entering data
- jog keys that are used for scrolling through data presented in some windows
- ENTER key used to enter any keyboard data into the controller
- ESCape key that is used to return to the previous window
- SHIFT key used in conjunction with the Soft keys to access different modes of operation
- DELete key used to delete programs, program lines or data entries

AUTO MODE WINDOW

This is the default mode into which the controller powers up.

- the mode window will display **AUTO MODE**.
- the description of the currently selected program is displayed
- the current line of the program that will be executed once the machine is in cycle or if the machine is incycle then it will indicate the current program line being executed.
- the current position of the axes will be displayed
- the current status of the auxiliary outputs
- all keys other than the Soft keys are disabled
- the Soft key options which are

Soft key 1	(PROG) to select the program mode (password may be required)
SHIFT and Soft key 1	(VER) to display the Software version installed
Soft key 2	(JOG) to select the axes jog mode (password may be required)
SHIFT and Soft key 2	(DATUM) to select the datumming mode (password may be required)
Soft key 3	(START) to start a cycle
Soft key 4	(LIST) to list all the programs resident in the controller (password may be required)
SHIFT and Soft key 4	(NEW) to create a new program (password may be required)

- the status bar will indicate that the current state of the machine e.g., out of cycle (i.e. stopped).

The selected program will run when Soft key 3 (START) is pressed providing that the ***MultiStop*** ***Plus*** has first been datummed . The status line will indicate that the program is running and the current program line being executed will be displayed as indicated in Figure 1.

When a program is being executed, pressing Soft key 3 (HALT) will halt the program in cycle, pressing START again will re start the program from where it halted. Soft key 4 (STOP) will terminate the program at the point of the STOP key press.

Pressing ESC will also HALT the execution of the program.
In the HALTED state pressing ESC will terminate the program.

PROGRAM WINDOW

This mode is selected by pressing Soft key 1 (PROG), a password may be required. In this mode a list the currently selected program that is resident in the controller will be displayed.
The title of the currently selected program is displayed.
For more detailed instructions on creating a program see section 0.

The following commands can be used to create a program,

- move an axis
- specify the feed rate (X or Y) for the move
- specify either incremental or absolute moves
- create a wait condition
- configure an axis as a slave axis
- synchronize the slave axis to the master axis

- set auxiliary outputs ON, OFF or ON for 100 ms
- assign any output to become a programmable limit switch
- assign any input to one of the internal variables

- write and call a subroutine

- modify the value of one of the internal variables
- perform mathematical functions on the internal variables

- write conditional commands

All of these commands are all available from the SOFT keys displayed in the program window.

The program window will consist of,
- the header field will display the program title.
- the right hand side will indicate which soft keys functions are currently available

The other keys that are active are,
- ESCAPE will return to the AUTO window, if no editing box is open, (see below)
- the UP and DOWN arrow keys are active
- ENTER will allow the current line of the program to be edited or saved.

VERSION WINDOW

This window is selected by pressing SHIFT and Soft key 1 (VER). The software version number and other firmware information is provided. This information may be requested when contacting the manufacturer for technical support.

Also in this window the facility exists to adjust the LCD contrast using the up and down arrow keys.

JOG MODE WINDOW

This mode is selected by pressing Soft key 2 (a password may be required). In the JOG mode

- JOG mode is indicated in the current window line
- ESCAPE will return to the AUTO window
- READY TO JOG (X) will be indicated in the status bar line

- all keys other than the JOG keys are disabled
- the Soft keys allow
    - the axis to be jogged to be selected
    - the jog speed to be set to coarse or fine feed
    - the datum window to be selected (password may be required)

DATUM MODE WINDOW

This mode is selected from either the AUTO or JOG windows (a password may be required), or by one of the external input lines.
In the DATUM mode

- DATUM mode will indicated on the screen
- ESCAPE will return to the AUTO window
- all keys are disabled
- the Soft key options are
    (DATUM X) which will begin an X datum cycle
    (DATUM Y) which will begin an Y datum cycle on two axis systems
- the status bar will indicate that the status of the datuming procedure

Pressing DATUM X soft key will commence the X datum cycle.
Pressing DATUM Y soft key will commence the Y datum cycle.
The datum cycles can be stopped at any time by pressing the STOP key.
The datum cycles can also be started and stopped using the external input lines.
.

LIST MODE WINDOW

This mode is selected by pressing Soft key 4 (LIST) and will list the directory of programs currently resident in the ***MultiStop*** Plus (a password may be required).

- The header field will display PROGRAM DIRECTORY.
- ESCape will return to the AUTO window
- Soft key 1 can be used to copy a program (the copy of the file that is created will carry a number suffix starting at zero for the first copy)
- Softkey 2 can be used to change the identity number of the program
- Softkey 3 can be used to change the description and title of the file.
- the UP and DOWN arrow keys are active

All programs generated on the ***MultiStop*** Plus controller will automatically be given numbers. Programs have associated numbers beginning at program number 0. Subsequent programs will be given the numbers 1, 2, 3 etc.

The selected program (i.e., the one that will be executed in AUTO mode and edited in PROGRAM mode) will initially be displayed in reverse video (i.e., blue text on a white background).
A different program can be selected by using the UP or DOWN arrow keys and then pressing ENTER.

The number of programs that can be stored in the controller depends on the length of the individual programs. For example, a maximum of 300 programs each containing approximately 60 lines can be accommodated. The maximum number of lines for each program is 100 lines in any one program. If the internal memory limits are exceeded an error will be generated.

A program can be deleted by selecting the program and pressing the DELete key.

To change the name of a program press <SHIFT> <SOFTKEY 3>. A name box will appear and by using either the UP or down arrows letters can be inserted into the name of the program. Upper or lower case letters can be selected. The numeric keys function as normal. Any letter or number can be edited by selecting it with the side arrow keys and then selecting a new character/number. Press ENTER when the correct description has been entered.

To change the description of a program press <SOFTKEY 3>. A description box will appear and by using either the UP or down arrows letters can be inserted into the description of the program. Upper or lower case letters can be selected. The numeric keys function as normal. Any letter or number can be edited by selecting it with the side arrow keys and then selecting a new character/number. Press ENTER when the correct description has been entered.

To change the identity number of a program press <SHIFT> <SOFTKEY 2>. A ID box will appear and the numeric keys can be used to assign a 2 digit number to the program. When reassigning ID numbers it is sometimes necessary to assign to a program an ID number that is already used. In this case first reassign the program with the lower ID number to a high unused ID number and then allocated the number just vacated to the required program.

GENERAL METHOD OF OPERATION

Normally the controller is configured such that after power up all datum positions will be lost. If this is the case then each axis must be datummed in turn.

DATUM MODE WINDOW

This mode is selected by pressing SHIFT and Soft key 2. A password may be required.
The Datum mode can also be selected using an external input.

- DATUM mode will indicated on the screen
- ESCAPE will return to the AUTO window
- all keys other than the Soft keys are disabled
- the Soft key options are
  - Soft key 3 (DATUM X) which will begin an X datum cycle
  - Soft key 4 (DATUM Y) which will begin an Y datum cycle on two axis systems
- the status bar will indicate that the status of the datuming procedure

AUTOMATIC DATUM CYCLE (if configured)

Pressing DATUM X soft key will commence the X datum cycle.
Pressing DATUM Y soft key will commence the Y datum cycle.
The datum cycles can be stopped at any time by pressing the STOP key or by asserting the STOP external input line.
A datum cycle can also be commanded using the external input lines. First the datum mode can be selected and then the external START line can be used to commence the datum cycle for the X axis.
Once the X datum cycle has been completed a second external START signal will begin the Y datum cycle.
The cycle is as follows,

- move the axis in the direction configured until the NEAR HOME signal is seen.
- reverse direction and move at slow speed until the reference marker pulse is seen.
- stop the axis and load the datum value into the axis position
- set the axis datummed flag

MANUAL DATUM (if configured)

To datum manually select JOG mode and jog the machine to the position required.
Then select the datum mode as described above.
Pressing DATUM X soft key will datum the X axis, and pressing DATUM Y soft key will datum the Y axis.
If the external lines are used then once datum mode has been selected an external start signal will datum **both** axes at the same time.

JOGGING

Each axis can be jogged around the machine by first selecting the JOG mode (a password may be required) and then use the UP/DOWN arrows to move the axes. The jog speed can be changed from FINE FEED to COARSE FEED at any time.

RUNNING A PROGRAM

To operate an automatic cycle, a program that describes that cycle must exist and the operator should select which program the unit is to execute from the LIST window.

On dispatch from the factory there will be an example programs installed in the controller. This file illustrates how programs can be constructed.

If a program exists in the controller then it can be selected for execution either using the keyboard using LIST or via the external digital input lies or via the RS 485 link.

Programs can also be downloaded from / uploaded to / the controller (via the RS485 link) to a PC running the appropriate software.

Once a program has been selected the cycle can now be started.

After the cycle has been started the cycle can be either

- stopped. The cycle will terminate immediately and the next cycle start will begin the program from the defined START position in the program
- paused (Halt). The cycle will stop at the current position and the next cycle start will begin execution from this point in the program. Any dwell instructions will time out.

RS485

If a suitable PC is connected to the ***MultiStop*** RS485 port then ***MultiStop*** will respond to the following commands,

upload configuration parameters
upload stored program names and ID number
upload program number
download configuration parameters
download program number
select program number
upload position data

AXES CONFIGURATION

Each axis on Multistop can be configured to be either a linear or rotary axis.

LINEAR MOVES

If the axis is a linear axis then the program associated for that axis will allow the user to program absolute or incremental moves in either direction up to a limit of 999999.99 mm..

ROTARY MOVES

When rotary movement for an axis is specified there is a difference in how the controller executes linear and absolute moves.

ROTARY INCREMENTAL MOVES

A move can be programmed in exactly the same way as for a linear axis, but the display is reset,

- to 0.000 every time the displayed value is greater than 360 degrees and a revolution counter (N. displayed close to the axis indicator) is incremented by 1.
  Thus if the machine is at 0.000 and a move of +766 is programmed, then when the axis reaches its target the display will indicate N = 2 and X = 46.000
- to 360.000 every time the displayed value is less than 0 degrees and a revolution counter (N. displayed close to the axis indicator) is decremented by 1
  If the machine is at 0.000 and a move of -766 is programmed, then when the axis reaches its target the display will indicate N = -2 and X = 314.000

ROTARY ABSOLUTE MOVES

A move greater than 360.000 cannot be programmed. However a signed absolute move can be programmed with the sign programmed in the instruction indicating the direction in which the axis will reach its target position

- a move of +40 will cause the rotate in the positive direction until the axis is at 40.000 ^{O}. If the starting position of the axis when the move instruction is executed is more than 40 ^{O} then the axis will rotate in the positive direction pass through 0 and stop at 40 ^{O} . The revolution counter will be incremented by 1
- a move of -40 will cause the rotate in the negative direction until the axis is at 40.000 ^{O}. If the starting position of the axis when the move instruction is executed is less than 40 ^{O} then the axis will rotate in the negative direction pass through 0 and stop at 40 ^{O} . The revolution counter will be decremented by 1

COMMANDS AND METHOD OF PROGRAMMING

There are two sets of programming commands that can be used on all Multistop units,

1. BASIC
2. ADVANCED

The method of programming below describes the use of the basic commands but the advanced commands can be used and inserted into the program in exactly the same way.
To edit the currently selected program press Soft key 1 (PROG) from the AUTO window. This will enter the editing mode.

If a new program is to be generated, then from the AUTO window press SHIFT Soft key 4 (NEW). The next available program reference will be assigned to this program and a program template will be opened and program editing mode selected. This template will show the program number in the top left of the display and include the following lines of program;

START
END_MAIN
END_FILE

with the START line highlighted.

A program can consist of a maximum of 100 lines.

When editing is complete press ESCape to exit and save the program. If any errors are found in the program a window will pop up to inform the user a problem exists. A list of warnings and errors are detailed in section 0. Error messages will not allow you to exit and save the program until the errors have been corrected. Warning messages can be ignored by pressing ESCape a second time.

TERMINOLOGY

MODAL COMMANDS

These commands, once set in a program remain valid until they are reset or changed by subsequent commands. Examples include FEED and MOTION commands.

TRUE / FALSE

ASSERTED

The asserted state of an input or an output can be made high or low.
For an input the asserted state is defined as the state which should cause an action to occur, for example if external start line is asserted high, then when that line is made high a cycle will start.
For an output the asserted state is defined as the state of the output line when a condition is true. For example if the inposition line is asserted high, then when the axis is inposition the line will be high, and low when the axis is not inposition.

VARIABLES

In the use of the commands it is necessary to understand the difference between a integer variable and a floating point variable.

INTEGER VARIABLES

These are variables (parameters that are used in a program) that are whole numbers for example 3 is an integer and 3.0 is not.
Normally integer variables could be quantities such as the number of times you wish to perform an action, e.g. The number of times you want to go around a program loop.

FLOATING POINT VARIABLES
These are variables (parameters that are used in a program) that are numbers which may have a fractional part, for example 68.000 is an floating point number and 68 is not.
Normally floating point variables could be quantities such as the target position you wish to move to action, e.g. X target = 123.985 or X = -345.954

BASIC COMMANDS

MOTION COMMAND

Used to specify whether a move is an ABSOLUTE move (i.e. a positional move relative to the zero point, for example - move to the position at 100mm) or an INCREMENTAL move (i.e. move an additional amount from the current position.
This command is modal.

If no motion command is inserts the program will assume that the required move is an absolute move, until it is changed by a specific incremental motion command. Thereafter all moves will be incremental unless commanded to be absolute by a new MOTION (X, ABS) command. Restarting the program will cause the exact same sequence of moves to be executed.

This command can be inserted into the program by pressing the soft key MOTION which causes the line MOTION(X, ABS) to be inserted above the highlighted line as shown

START
MOTION (X, ABS)
END_SUB

The command can be altered by pressing the right arrow or enter key which will open up an editing box

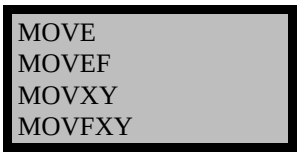
Select one axis using the UP /DOWN arrow keys and press enter. The second editing box opens which allows either absolute or incremental motion to be selected,

When the appropriate choice has been made press ENTER to insert the command in the program

START
MOTION (Y, INC)
AUX(1,ON)
MOVXY (255.450, -100.550)
MOVE (Y, -90.000)
etc.

MOVE COMMANDS

The various move option that can be selected are,

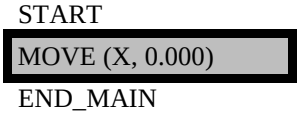


1. MOVE COMMAND

2. MOVEF COMMAND

Used to effect a movement of an axis.

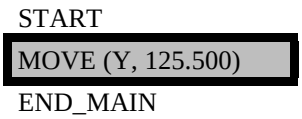
These (MOVE and MOVEF) have identical functionality with the exception that the move command is programmed with a positional value e.g. MOVE(X, 10.95) and the MOVEF uses one of the floating point variables as the target position, e.g., MOVEF(X, FLT1). The value of the variable can be set or calculated within the powertalk program, see example at the end of this section.
The text below will give examples of the MOVE command but the MOVEF command could equally be used wherever a MOVE is written.
The MOVE command can be inserted into the program by pressing the soft key **(MOVE) when** the line MOVE (X, 0. 000) will be inserted and highlighted as shown



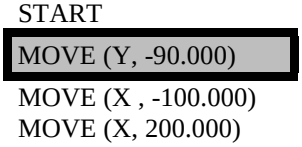
The move command can be edited by pressing the right arrow or enter key which will open up an editing box

allowing X or Y to be selected using the UP or DOWN arrow key. Select Y for this example. Press ENTER to confirm the selection and the editing box will prompt the programmer to enter a target for the move

Enter an appropriate value say 125.500 and press ENTER. The data is inserted into the program and the editing box is closed.



If another move is required repeat the above several times and the new move instructions will be inserted ABOVE the highlighted line. Edit as required.



MOVE (Y, 125.500)
END_MAIN
END_FILE

The move command can be edited by pressing the right arrow or enter key which will open up an editing box again

Any instruction in the program (except for the template lines) can be removed by pressing DEL. Any editing can be aborted by pressing ESC.

Example of the use of the MOVEF command.

Set FLT1 to 12.56 using the float command,
FLOAT(FLT1,12.56)
The X axis can be moved to this position using the command
MOVEF(X, FLT1)
The value of FLT1 can be altered as required within the program.

3. MOVXY COMMAND

4. MOVFXY COMMAND

Used to effect the simultaneous movement of 2 two axes.

These command are not available on single axis units.
There commands (MOVXY and MOVFXY) can be used to effect movements of both axes simultaneously. These have identical functionality with the exception that the MOVXY command is programmed with a positional value e.g. MOVXY(99.23, 10.95) and the MOVFXY uses two of the floating point variables as the target position, e.g., MOVFXY(FLT2, FLT1). The values of the floating point variables can be set or calculated within the powertalk program providing the advanced programming features are enabled..
The text below will give examples of the MOVXY command but the MOVFXY command could equally be used wherever a MOVXY is written.

Selecting the MOVXY command causes the line MOVXY (0.000, 0. 000) will be inserted as shown

START
MOVXY (0.000, 0.000)
END_MAIN
END_FILE

The command can be edited by pressing the right arrow or enter key which will open up an editing box to edit the X value

Enter an appropriate value say 255.450 and press ENTER. The data is inserted into the program and the editing box now contains the Y value.

Enter the numeric value required, say -100.550. Press ENTER to accept the data. The program will then contain this data.

START
MOVXY(255.450, -100.550)
END_MAIN
END_FILE

5. MOVE AXIS UNTIL INPUT ASSERTED

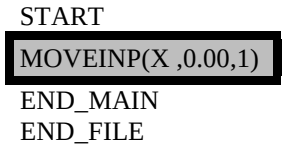
Used to move an axis towards a target position. The movement will continue until either the target is reached OR the specified input is asserted.

The various option that can be selected are,

The commands are MOVEINP , MOVEINPF can be used to effect either an incremental or absolute move. The commands have identical functionality with the exception that the move command is programmed with a final target position for X e.g. MOVEINP(X, 20.85, 2) , the MOVEINPF can use one of the floating point variables as the target position, e.g., MOVEINPF(X, FLT1, 2). The parameter 2 in these examples refer to input 2. Thus in the first example if the command is executed the X axis will move towards the position 20.85 and will either reach that position or else the move will be terminated by a input 2 becoming valid.

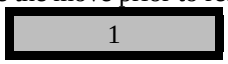
The text below will give examples of the MOVEINP command but the MOVEINPF or the MOVXYINP commands could equally be used wherever a MOVEINP is written.

The MOVEINP command can be inserted into the program by pressing the appropriate soft key 1, when the line MOVEINP(X, 0.000,1) will be inserted and highlighted as shown,

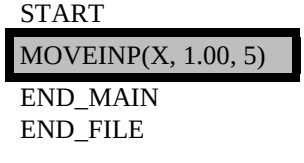


The MOVEINP command can be edited by pressing the right arrow or enter keys which will open up an editing box

allowing X or Y to be selected using the UP or DOWN arrow key..
Select X for this example. Press ENTER to confirm the selection and the editing box will prompt the programmer to enter a target position for the move. Enter an appropriate value say 1.000 and press ENTER. The data is inserted into the program and the editing box will prompt the programmer to enter an input number to terminate the move prior to reaching the target position



Enter an appropriate value say 5 and press ENTER. The data is inserted into the program and the editing box is closed



Note that no feed has been specified for these moves. This must be done before the program can run. To enter feed values see below.

FEED COMMANDS

Used to specify the rate at which the axis will move in the steady state (mm/min). The command is modal.

The various options that can be selected are,

FEED
FEEDF

6. FEED

7. FEEDF

These commands allows the programmer to specify the feed rate at which the axes are to be moved in conjunction with the MOVE commands.
These commands have identical functionality with the exception that the FEED command is programmed with a numeric value e.g. FEED (X, 10.00) and the FEEDF command uses one of the floating point variables as the feed value, e.g., FEEDF(X, FLT1). The value of the variable can be set or calculated within the powertalk program.

The text below will give examples of the FEED command but the FEEDF command could equally be used wherever a FEED command is written.

The FEED command must be inserted into the program before the move command by selecting one of the FEED commands, FEED for example, which causes the line FEED(X ,0.000) to be inserted above the highlighted line as shown,

```
START
MOTION (Y, INC)
FEED(X,0.000)
MOVE (Y, 255.450)
END_MAIN
END_FILE
```

The FEED command by be edited by pressing the right arrow or enter keys which will open up an editing box

X
Y

allowing X o to be selected using the UP or DOWN arrow key.

Select Y for this example. Press ENTER to confirm the selection and the editing box will prompt the programmer to enter a feed rate, say 1.000 . Now press ENTER. The data is inserted into the program and the editing box is closed.

```
START
FEED (Y, 1.000)
MOVE (Y, 255.450)
END_MAIN
END_FILE
```

Once the feed rate for the axes has been set it will remain valid for all subsequent moves until changed by the program.

WAIT COMMANDS

Commands that can be used to pause a pause in the program

The various options that can be selected are,

HALT
WAIT
DWELL
DWELLF

8. WAIT COMMAND

Used to hold the program at the point at which the command is inserted and to remain at that point until the input specified in the command is asserted.

This command can be applied to any of the 24 inputs.

When this command is inserted into the program the line WAIT(0) appears above the highlighted line,

START
MOTION (Y, INC)
AUX(1,ON)
WAIT(0)
MOVXY (255.450, -100.550)
etc.

The WAIT instruction can be given an input number by pressing the enter key which will open up an editing box

0

Enter the input number to be tested making sure that the number corresponds to the input line that the controlling signal is connected to, say input line 4, and press enter

START
MOTION (Y, INC)
AUX(1,ON)
WAIT(4)
MOVXY (255.450, -100.550)
etc.

the program will switch auxiliary output 1 on and then wait until input 4 is true and then begin the MOVXY instruction..

9. HALT COMMAND

Used to halt the program at the point at which the command is inserted .
The program can be restarted from this point by either a keyboard START or an external START signal.

When this command is inserted into the program the line HALT will be inserted above the highlighted line,

START
MOTION (Y, INC)
MOVXY (255.450, -100.550)
HALT
etc.

10. DWELL

11. DWELLF

Used to introduce a timed pause in the program at the point at which the command is inserted.

There are two versions of this command, DWELL and DWELLF. These have identical functionality with the exception that the DWELL command is programmed with a numeric value e.g. DWELL(10.00) and the DWELLF command uses one of the floating point variables as the feed value, e.g., DWELLF(FLT1). The value of the variable FLT1 can be set or calculated within the powertalk program.
The text below will give examples of the DWELL command but the DWELLF command could equally be used wherever a DWELL command is written.
When the DWELL command is inserted into the program the line DWELL (0.000) will appear above the highlighted line as shown,

START
MOTION (Y, INC)
AUX(1,ON)
DWELL (0.000)
MOVXY (255.450, -100.550)
MOVE (Y, -90.000)
etc.

The command can be edited by pressing the right arrow or enter key which will open up an editing box,

0.000

A time in seconds can be entered and then press enter to insert the command into the program.

START
MOTION (Y, INC)
AUX(1,ON)
DWELL (0.500)
MOVXY (255.450, -100.550)
MOVE (Y, -90.000)
etc.

OUTPUT COMMAND

Used to set or reset the outputs allocated to the auxiliary function signals

12. SET / RESET AUXILIARIES

Used to control the auxiliary outputs to during a program.

Selecting the AUX option after pressing the OUTPUTS soft key causes the line AUX (1, ON) to be inserted as shown

START
AUX (1, ON)
MOVXY (255.450, -100.550)
etc.

The auxiliary command by be edited by pressing the enter key and selecting a number (say 5), then press ENTER and the menu box will change to

OFF
ON
PUL

which allows the auxiliary to be turned ON, OFF or set to come on for a fixed period (pulsed for 0.5 Second). The UP / DOWN arrows will select one state. When the correct state has been set press ENTER to accept the data. The program will then contain this data.

START
AUX (5, ON)
MOVXY (255.450, -100.550)
etc.

Unless canceled elsewhere in the program this auxiliary will now remain ON.
It can be reset, if required by inserting another AUX command in the program at an appropriate position,

START
AUX (5, ON)
MOVXY (255.450, -100.550)
AUX (5, OFF)
MOVE (Y, -90.000)
etc.

The auxiliary will now be on during the MOVXY(255.450, -100.550) instruction and will be switched off before commencing the MOVE (Y, -90.000) instruction.

If several auxiliaries are required to be on during a move they must all be set individually to come and go off at the appropriate places.

To keep an auxiliary on for a given period of time, the AUX and DWELL commands must be used. For example for a 300 ms period auxiliary 7 will come on before the instruction MOVXY (255.450, -100.550) begins

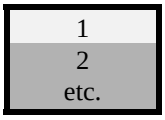
START
AUX (7, ON)
DWELL (0.3)
AUX (7, OFF)
etc.

13. PLS COMMAND

Used to control the state of the auxiliary outputs (depending on axis position) during a cycle.

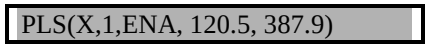
Each output can be configured as a programmable limit switch (PLS)
If an output is configured as a PLS then this command can specify at which positions it is to be turned on and off. It also allows the programmer to specify which axis is the controlling axis and also to enable or disable the PLS function at selected points in the program. Once an output is configured as a programmable limit switch it cannot be used as a conventional auxiliary output as described in the section SET / RESET AUXILIARIES
Selecting the PLS option after pressing the OUTPUTS soft key causes PLS(1,X,ENA,0.00,0.00) to be inserted and highlighted as shown,

The PLS command can be edited by pressing the ENTER key which will open up an editing box for the controlling axis



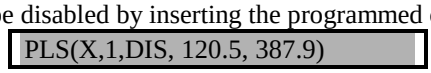
allowing the output number to be specified, (any of the auxiliary outputs can be used). The next ENTER key press will allow the X or Y axes to be selected as the controlling position for the programmable limit switch.. Press ENTER to confirm the selection and a new editing box will open to allow the programmer to enable or disable the PLS command,

Press ENTER and then set the position at which the PLS output is to be switched ON (asserted) and finally the position at which the PLS output is to be switched OFF. The final instruction may look as follows



In this example output 1 will always be ON if the X position lies between 120.5 and 387.9 mm.

The function can be disabled by inserting the programmed command



at the appropriate position in the program. Note that when disabling a PLS instruction the switching positions are of no consequence.
Note that providing the PLS output is not disabled it will ALWAYS remain active providing Multistop is in cycle. It is also possible to configure the asserted state of the output to either HIGH or LOW.

SUB COMMAND

The various options that can be selected are,

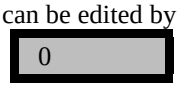
14. CALL

Used to call a subroutine from the point in the program at which the instruction is inserted.

It causes the line GO_SUB(0) to be inserted above the highlighted line as shown

START
GO_SUB (0)
etc.

The call command can be edited by pressing the RIGHT arrow which will open up an editing box



Enter the subroutine number applicable and press ENTER.

It is essential to write a subroutine with the same number see section SUBROUTINE

15. SUBROUTINE

Used to write a subroutine. The subroutine must be identified with an unique number that corresponds to the CALL command that requires the use of this subroutine.

A maximum of 9 nested subroutine calls are permitted.
Within the subroutine all programming commands can be used to write the subroutine program.
This command can be inserted into the program by pressing SHIFT Soft key 3 (SUB) which causes the lines SUB(0) and END_SUB to be automatically inserted underneath the END_MAIN line of the program. If several subroutines exist the new subroutine will be inserted at the lowest position possible in the program.

END_MAIN
SUB (0)
END_SUB
END-FILE

The subroutine can be given a number by pressing the right arrow or enter key which will open up an editing box

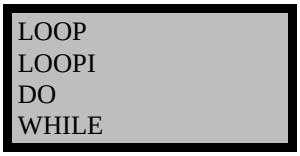
Enter the number of the subroutine making sure that the number corresponds to the number used in the CALL command (see section 0). Insert any programming instructions that are appropriate within the subroutine. For example,

```
SUB(1)
   MOVE(X, 100.00)
   AUX (1,PULSE)
   MOVE(Y, 20.00)
END_SUB
```

LOOP COMMANDS

Used to repeat a section of program until a predefined condition has been satisfied

The various options that can be selected are,

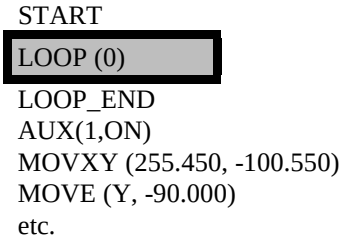


16. LOOP

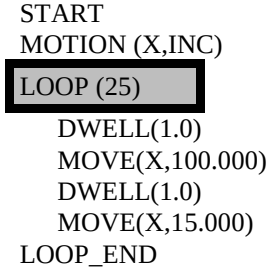
17. LOOPI

Used to repeat a section of program a given number of times.

These commands allow the program to execute all of the lines contained between the start LOOP command and the LOOP_END command, a given number of times.
These commands have identical functionality with the exception that the LOOP command is programmed with a numeric value e.g. LOOP (10) and the LOOPI command uses one of the integer variables as the loop value, e.g., LOOPI(INT1). The value of the variable INT1 can be set or calculated within the powertalk program.
The text below will give examples of the LOOP command but the LOOPI command could equally be used wherever a LOOP command is written. A maximum of 9 nested loops is permitted.
The LOOP command can be inserted into the program by pressing Soft key 3 (LOOP) which causes the lines LOOP(0) and LOOP_END to be inserted above the highlighted line as shown



Any of the programming commands can now be used within this loop. Thus if the machine was required to reciprocate between X =15.000 and X =100.000 a total 25 times (with a delay of 1 sec at each end of travel) then the following instructions would be written,



18. DO COMMAND

Used to loop around a program section until an internal variable becomes false.

In the following example the variable say INT3 may have been assigned to follow the state of input 4.

When this command is inserted into the program the lines DO and DO_WHILE(INT 0) will be inserted above the highlighted line as shown,

```
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT3,4)
DO
DO_WHILE(INT 0)
etc.
```

Program instructions can be placed within this DO and DO_WHILE loop

Specify the INT number associated with the DO_WHILE statement by pressing the enter key and selecting the number required (say 3)

```
START
DO
/*      place any program code that is suitable for the application here      */
DO_WHILE(INT 3)
etc.
```

the program will the continue to execute the lines inserted within the DO and DO_WHILE loop while the value of integer variable INT3 and thus input #4 **remains** true .

Note that the code within the DO loop is **ALWAYS EXECUTED AT LEAST ONCE**.

19. WHILE COMMAND

Used to loop around a program section while an internal variable is true (not zero).

The variable say INT 5 may have been assigned to follow the state of input 21.

This command can be inserted into the program pressing SHIFT and Soft key 1 on menu 3 (WHILE) which causes the lines WHILE(INT 0) and WHILE_END to be inserted above the highlighted line as shown,

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT 5,21)
WHILE(INT 1)
WHILE_END
etc.
```

Program instructions can be placed within this WHILE loop

Specify the INT number associated with the WHILE statement (INT5 for example) using the arrow keys and press enter once more.

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT5,21)
WHILE(INT5)
/*      place any program code that is suitable for the application here      */
WHILE_END
etc.
```

the program will then continue to execute the program inserted within the WHILE(INT5) and WHILE_END loop as long as the value of input #21 (and hence INT5) is true (not zero)

INPUT COMMAND

Used to assign the state of any of the input lines to one of the internal integer variables .

The integer variable e.g., INT 1 can then be used as a test with the IF, DO and WHILE commands. This command causes the line INPUT(INT1,0) to be inserted above the highlighted line as shown,

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT1,0)
MOVXY (255.450, -100.550)
etc.
```

To specify the integer variable number that is to be assigned to the external input line, for example input line 8.

```
INPUT(INT1,8)
```

INT1 will now always reflect the state of input line 8 and this can be used to within the program to perform conditional tasks.

IF COMMAND

Used to execute one of two sets of instructions depending on the state of one of the internal integer variables.

For this example use internal variable INT6 and input line 11.

This command can be inserted into the program by pressing and Soft key 2 on menu 3 (IF) which causes the lines IF(INT1), ELSE and END_IF statements to be inserted above the highlighted line as shown,

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT6,11)
IF(INT1)
ELSE
END_IF
etc.
```

Program instructions can be placed above and below the ELSE statement.

Specify the INT number associated with the IF statement by selecting the number you wish you use (6 for example) and press ENTER once more.

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT6,11)
IF(INT6)
/*      place any program(A) code that is suitable for the application here      */
ELSE
/*      place any program (B) code that is suitable for the application here     */
END_IF
etc,
```

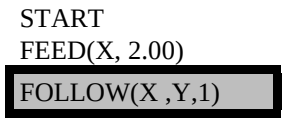
the controller executes program A if input 11 is true (not zero) or else it executes program B.

FOLLOW COMMAND

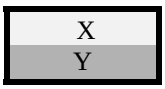
Used to slave one axis to another (or an external axis).

Its format is FOLLOW(Master, Slave, Gear Ratio).
Thus for example FOLLOW(X, Y,2.0) would cause the Y axis (slave) to follow the X axis with a gear ration of 2.0. In this example the slave axis would always follow the master axis but the slave's movement would always be half of the masters movement, so as the X axis moves to 100.00 the slave would move to 50.00.
The follow command can be disabled within a program if required by setting the gaer ratio to 0.00

The FOLLOW command can be inserted into the program by pressing the appropriate soft key, when the line FOLLOW(X, Y,1) will be inserted and highlighted as shown,



The FOLLOW command by be edited by pressing the right arrow or enter keys which will open up an editing box for the master axis



allowing X or Y to be selected using the UP or DOWN arrow key. Once the master has been set the slave is automatically set as the other axis
Select X for this example. Press ENTER to confirm the selection and the editing box will prompt the programmer to enter a gear ratio for the slave axis

Enter an appropriate value say 2.000 and press ENTER. The data is inserted into the program and the editing box is closed. All further programmed move instructions for the slave axis will be ignored until the follow command is terminated. This can be done by setting a gear ration of 0 in a new FOLLOW command.

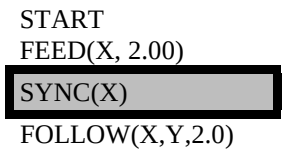
If the master axis is not controlled by Multistop then providing the encoder signals from the master axis are connected to the encoder inputs assigned by the Multistop program as being the master (X in the above example) then the slave axis (Y) will follow the external axis as specified in the FOLLOW instruction.

It is often necessary to synchronize the two axes. If both are being controlled by Multistop then it is possible to perform datum cycles that will result in the axes being synchronized correctly. If the master axis is an external axis then the SYNC command can be used to achieve synchronization.

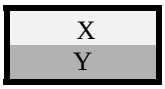
SYNC COMMAND

Used to synchronize an external master axis and internal slave axis.

It cause the program to halt at the position of the command and to wait at that point until the marker (fiducial) pulse of the external axis is asserted. On seeing the marker pulse the program continues to the next instruction which should always be the FOLLOW command.
Pressing the SYNC soft key causes SYNC(X,) to be inserted and highlighted as shown,



The SYNC command by be edited by pressing the right arrow or enter keys which will open up an editing box for the master axis

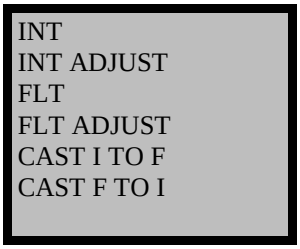


allowing X or Y to be selected using the UP or DOWN arrow key. Press ENTER to confirm the selection and the editing box will close. The slave axis (Y) will now begin to follow the X axis only when the X marker pulse has been asserted.

VARIABLE COMMANDS

Used to manipulate the internal (both integer and floating point) variables.

The various options that can be selected are,

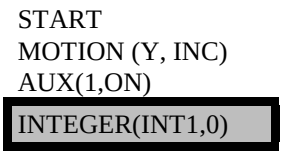


20. INT COMMAND

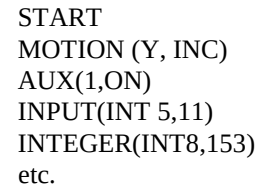
Used to set an integer variable to a given value

For this example use internal variable INT7 and a value of 153.

The command can be inserted into the program by pressing and Soft key 3 on menu 3 (INT) which causes the line INTEGER(INT1,0) to be inserted above the highlighted line as shown,



Specify the INT number associated with the INTEGER command by enter key and selecting 8 and press ENTER once more. It is then possible to enter the value which is to be assigned to INT8 (the range of the value is +32767 to - 32767). Enter 153 and press ENTER once more.



This value can be used with the ADJUST command, see below.

21. ADJUST COMMAND

Used to increment or decrement one of the integer or floating point variables by a specified amount.

The choice of commands are,
INT ADJUST AND FLT ADJUST.

Selecting the INT ADJUST option inserts the program line ADJUST(INT1,0) above the highlighted line as shown,

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT8,11)
ADJUST(INT1,0)
etc.
```

For this example use internal variable INT8 and an increment of +2.

Specify the INT number associated with the ADJUST statement by pressing enter key and select the number you wish to use (8 for example) using the arrow keys and press enter once more. It is then possible to enter the value by which the variable will be adjusted. Enter a value of 2 and press ENTER once more.

```
START
MOTION (Y, INC)
AUX(1,ON)
INTERGER(INT8,11)
ADJUST(INT8, 2)
etc.
```

In this program the value of integer variable 8 will have 2 added to its original value after the adjust statement. Positive or negative adjustments to integers can be made in this way.

The commands DO_WHILE(INT x) and WHILE(INT x) can be used with this command to change the program flow.

NOTE: When adjusting a variable to use with the program flow statements ensure that it is possible for the variable to become zero. Any non zero number is true, only a variable that is at zero is false.

22. FLT COMMAND

Used to assign a number to a floating-point (real number) variable.

FLOAT (FLTn, 0.00) set floating point variable n to any signed floating point

If no value is assigned to the floating point variable it is set to 0.000.
For this example use internal variable FLT7 and a value of 153.56.
The command can be inserted into the program above the highlighted line as shown,

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT 7,11)
FLOAT(FLT1,0.0)
etc.
```

Specify the FLT number associated with the FLOAT statement by pressing the enter key and selecting the number you wish you use (7 for example) . Then press ENTER once more. It is then possible to enter the value which is to be assigned to FLT7 (the range of the values is unrestricted). Enter 153.56 and press ENTER once more.

```
START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT 5,11)
FLOAT(FLT7,153.56)
etc.
```

READ COMMANDS

23. POSITION

Used to determine the axis position at any point within the program. This information is assigned to a selected floating point variable and can be used to determine the program flow

Press the READ soft key and select READ_POSITION from the dialogue box. Press ENTER and a line will be inserted in the program,

```
START
POSITION (X,FLT1)
END_MAIN
END_FILE
```

Press ENTER more and select X or Y as the axis to be read., enter the selection (say Y for example) and then select the floating point variable that is to be used to save the positional information, say FLT3. The program would then be,

```
START
POSITION(Y,FLT3)
END_MAIN
END_FILE
```

24. GET_TIME

Used to read the internal timer (in secs with a resolution of 10 ms) at any point within the program. This information is assigned to a selected floating point variable and can be used to determine the program flow

Press the READ soft key and select READ_TIME from the dialogue box. Press ENTER and a line will be inserted in the program,

```
START
GET_TIME(FLT1)
END_MAIN
END_FILE
```

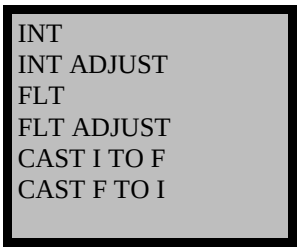
Press ENTER once more then select the floating point variable that is to be used to save the time information, say FLT6. The program would then be,

```
START
GET_TIME(FLT6)
END_MAIN
END_FILE
```

ADVANCED COMMANDS

VARIABLE COMMANDS

The various options that can be selected are,



The first four commands have been described above in the section on Basic commands.

25. CAST I TO F

CAST I TO F INT > FLT (INTn, FLTm) convert integer variable n to floating point variable m
For example , if INT n was 3 then FLTm would be 3.000

26. CAST F TO I

CAST F TO I FLT > INT (FLTn,INTm) convert floating point variable n to integer variable m
For example , if FLT n was 5.000 then INTm would be 5
if FLT n was 5.216 then INTm would be 5
if FLT n was 5.893 then INTm would be 6

MATH COMMANDS

Used to do arithmetic and number tests on variables from within the program.

The various math commands available are,

27. ABS

ABS (INTn, INTm)	place the modulus of integer variable n into integer variable m

28. FABS

FABS (FLTn, FLTm)	place the modulus of floating point variable n into floating point variable m

29. ADD

ADD (FLTn, FLTm, FLTp)	add floating point variables n and m and place result in floating point variable p

30. MINUS

MINUS (FLTn, FLTm, FLTp)	subtract floating point variables n and m and place result in floating point variable p

31. DIV

DIV(FLTn, FLTm, FLTp)	divide floating point variables n and m and place result in floating point variable p

32. MULT

MULT (FLTn, FLTm, FLTp)	multiply floating point variables n and m and place result in floating point variable p

These command allows the programmer to do arithmetic operations on two floating point variables and puts the result in a third floating point variable. The following example will use the ADD command
For this example use add the floating point variables FLT8 and FLT7 and place the result in FLT2
Selecting the command inserts ADD(FLT1, FLT2, FLT3) to be inserted above the highlighted line,

START
MOTION (Y, INC)
AUX(1,ON)
INPUT(INT8,11)
ADD(FLT1,FLT2,FLT3)
etc.

Specify the FLT number associated with the ADD command by pressing the enter key which will open up an editing box

FLT1
FLT2
FLT3
FLT4
FLT5
FLT6
FLT7
FLT8

Select the number you wish you use (8 for this example) using the arrow keys and press enter once more. It is then possible to enter the value associated with the second floating point variable, (7 for this example) and press ENTER once more to enter the value associated with the third floating point variable, (2 for this example) . Finally press ENTER once more to confirm the entry.

```
START
MOTION (Y, INC)
AUX(1,ON)
FLOAT(FLT8,11.0)
ADD(FLT8,FLT7,FLT2)
etc.
```

33. WHOLE

WHOLE (FLTn, INTm)	place the whole part of floating point variable n into integer variable m

34. FRAC

FRACTION (FLTn, FLTm)	place the fractional part of floating point variable n into floating point variable m

35. EQUAL TEST (=)

= (FLTn, FLTm, INTn)	make floating point variable m equal to floating point variable n

36. GREATER THAN TEST (>)

(FLTn, FLTm, INTn)	set integer variable n to 1 if floating point variable n is greater than floating point variable m and to zero if floating point variable n is less than or equal to floating point variable m

37. GREATER THAN TEST (> =)

> = (FLTn, FLTm, INTn)	set integer variable n to 1 if floating point variable n is greater than or equal to floating point variable m and to zero if floating point variable n is less than to floating point variable m

38. LESS THAN TEST (<)

< (FLTn, FLTm, INTn)	set integer variable n to 1 if floating point variable n is less than floating point variable m and to zero if floating point variable n is greater than or equal to floating point variable m

39. LESS THAN OR EQUAL TEST (<=)

< = (FLTn, FLTm, INTn)	set integer variable n to 1 if floating point variable n is less than or equal to floating point variable m and to zero if floating point variable n is greater than floating point variable m

Any of these tests can be inserted into a program at any selected position by selecting the command required and pressing ENTER. Once the command is in the program the values on n and m for the floating point variables and the value n for the integer variable can be set to any of the allowed numeric values between 1 and 10. The result of the test causes the integer variable to be set to 1 if the test is true and to 0 if the test is false.

EXAMPLES

set integer 3 to -5 INTEGER (INT3, -5)

add 4 to integer variable 6 ADJUST (INT6, 4)

if the value integer 3 is -16 and it is required to place its modulus (i.e., 16) in integer variable 8
ABS(INT3, INT8)

if the integer variable 2 was 12 and it was required to convert this to a floating point number, i.e., 10.000 stored in floating point variable 5,
INT>FLT (INT2, FLT5)

set floating point variable to 3 to -5.12 IFLOAT (FLT3, -5.12)

add 4.23 to floating point variable 6 ADD_FLOAT (FLT6, 4.23)

if the value floating point 3 is -16.35 and it is required to place its modulus (i.e., 16.35) in floating point variable 8
ABS(FLT3, FLT8)

if the floating point variable 2 was 12.72 and it was required to convert this to a integer number, i.e., 13 stored in floating point variable 5,
FLT>INT (FLT2, INT5)

FRONT END

In certain applications it may be convenient to allow the operator to change certain variables in the program without allowing him to change the structure of the program. This can be done using a FRONT END window, in which the variables that are allowed to be changed are presented to the operator. Behind this front end window will be a standard power talk program that can be locked out from the operator and which requires a pass number to enter.
The front end allows parameters to be entered and passed to the powertalk program. The text associated with these parameters can be set to meaningful text in the configuration setup of the front end window.
As an example consider a simple cut to length application in which a 100 lengths of material of length 125mm and 200 lengths of 250 mm are required. It is also required to display the number of lengths cut at any time and to reset the number cut every time the cycle is started.

The parameters of interest to the program are assigned to program variables, either integers for whole numbers, or floating point variables for non integer numbers, i.e., any number that may have a decimal component. In the above example the number of sheets cut and the number of cuts required are all whole numbers and are thus integers. The lengths to be cut may not be a whole number and are thus floating point numbers.
The front end text may be as follows,

Number of first cuts done	0	(assigned to INT1)
Number of first cuts required	100	(assigned to INT2)
Length of first cut	125.000	(assigned to FLT1)
Number of second cuts done	0	(assigned to INT3)
Number of second cuts required	200	(assigned to INT4)
Length of second cut	250.00	(assigned to FLT2)

The operator may edit any of these values.
The powertalk program would use these variables to perform the task requested. The program may be as follows,

```
START
        INTEGER(INT1,0)         ;reset no of first cuts done
        INTEGER(INT3,0)         ;reset no of second cuts done
        MOTION(X,INC)           ;all moves to be incremental
        FEED(X,100)             ;set the feed rate for the move

        LOOPI(INT2)             ; do the loop 100 times (value set by INT2)
                MOVEF(X,FLT1)   ;move a distance equal to programmed first length
                ADJUST(INT1,1)  ;increment no of first cuts counter
                AUX(1,PUL)      ;pulse an output for a shear
                DWELL(2)        ;wait 2 sec
        ENDLOOP

        LOOPI(INT4)             ;do the loop 200 times (value set by INT4)
                MOVEF(X,FLT2)   ;move a distance equal to programmed second length
                ADJUST(INT3,1)  ;increment no of second cuts counter
                AUX(1,PUL)      ;pulse an output for a shear
                DWELL(2)        ;wait 2 sec
        ENDLOOP
END_MAIN
END_FILE
```

This front end can be set up as follows,

- in configuration enable the front end window
- assign the number of variables required to the window (max 10)
- assign the text required to each variable
- assign each variable to a numbered integer variable or to a numbered floating point variable.

ERROR MESSAGES

The following error messages can be displayed. Error messages cannot be ignored, warnings can.

Error Messages	**Cause**
System Disabled	System enable input is not present (disabled will be displayed on the right of the status bar)., or system enable input has not been assigned to an input.
Cannot start. Not datummed	The Multistop must either run a datum cycle or be datummed manually before a cycle can be run.
Call to undefined subroutine	A call to a subroutine has been programmed, but no subroutine exists.
No I/O power disconnected	The external 24V DC required for the inputs and outputs is not connected.
Program Full Delete Lines	The Maximum number of program lines per program is 100.
End Stop Hit	A Not End Stop input is low. The current move will stop and Jog mode will be selected.
Following Error to Solve	A following error occurs when the controller is commanding a move but there is not encoder feedback indicating a move is taking place. It can be caused for a number of reasons. 1. Faulty encoder or encoder cable. 2. Faulty servo drive (causing a motor move which has not been commanded) 3. Incorrect motor or encoder phase set in Configured Parameters (See the Technical Specification) After a following error the controller must be redatummed.
Warning Messages	
Feed rate not set	A new program defaults the feed rate to zero. If no feed rate has been programmed using the FEED command or if a zero feed has been programmed this warning will be displayed when exiting program mode.