Parker

Automation

CE

ViX250IE, ViX500IE, ViX250IH, ViX500IH, ViX250IM & ViX500IM

User Variables Addendum

November 2005

1. Introduction

Overview

The ViX user variable functionality permits the user to read and write up to 8 user-defined variables. These variables can be used to perform mathematical operations and also in combination with certain EASI code commands, where they can be used in selected data fields or in evaluation statements.

User variable data can be set directly by the user or control system, via the serial communications port or user program using the VAR command, or indirectly within a program structure by using the VMATH command. User variables may also be used in combination with the ViX system variables,

User variable data is only stored when a save (SV) command is executed. Following a reset or a power cycle, the user variable data will be returned to the values current at the time of the last save command. If no save command is executed prior to a reset, all user variable data will be lost.

Format

The eight user variables are labelled VAR0-VAR7, and are long-integer types with a range from -2,147,483,647 to +2,147,483,647. Binary and real (float) number types are not supported.

Standard mathematical functions of addition, subtraction, multiplication and division are possible. Where the result of an operation gives a non-integer value, the result is not rounded up or down but is simply truncated.

Example

214783647 / 4 = 536870911.75

The answer stored in the User Variable will be 536870911, not 536870912

Compatibility

The ‘user variables’ functionality is only available on the ViX models listed below, when used in combination with firmware version 3.00 or greater:

ViX250IE	ViX250IH	ViX250IM
ViX500IE	ViX500IH	ViX500IM

This functionality is not available on ViX base (analogue command) and CANopen units.

2. Command Reference

Command Description

Each command has a simple 1 to 7 character name, usually an abbreviation of its full descriptive title. Only commands applicable to the use of the user variables are listed in this addendum; please refer to the appropriate ViX user guide for full details on all other commands.

Each individual description will include a one-line header giving the abbreviated name followed by its full name. The following pages give the command syntax, units of measurement, range of values, any default value and a reference to other related commands. Where commands contain a list of parameters, a simple layout displays only the syntax of the command.

Every command requires an address. Where several drives need to respond to a common set of global commands, prefix each command with the address 0. To prevent spurious feedback any report or read command using address 0 will be ignored. Note a drive will ignore a command missing an address prefix. The address is identified in the command syntax by the prefixed character ‘a’.

Where commands (such as IF, R, TR, and W) include a system variable it is treated as a command parameter. System variables store internal drive values and settings. Each variable is capable of being read and tested, and some may be written to, but they are all dedicated for a particular use by the system and cannot be used for storing user data within a program.

This addendum is applicable to software revision 3.0 and onwards.

A Acceleration/Deceleration

Syntax	Units	Range of ‘n’	Default	See also
aAn	See SCALE	0.01 to 99999.99	10	AA AD SCALE VAR VMATH

Description This command will set both the acceleration and deceleration rates of the motor to the same value. Values set for the **AA** and **AD** commands are over-written, if previously set.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the acceleration and deceleration rates of axis 1 to 120 rps^2, type	**1A120**
To determine the acceleration of axis 1, type	**1A**
The response is	**120.0 120.0*
Overrange value	**1A505010**
Will be reported as	**E* (meaning error)

To set the acceleration and deceleration rates via a user variable value, type	**1W(VAR3,130)**
	1VMATH(A,=,VAR3)

AA Acceleration

Syntax	Units	Range of ‘n’	Default	See also
aAAn	See SCALE	0.01 to 99999.99	10	A AD SCALE VAR VMATH

Description The **AA** command will set or report the programmed linear acceleration rate of the motor. The acceleration value assigned to the **AA** command is over-written, if previously set.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the acceleration rate of axis 1 to 120 rps^2, type	**1AA120**
To determine the acceleration of axis 1, type	**1AA**
The response is	**120.0*
Overrange value	**1AA100002**
Will be reported as	**E* (meaning error)

To set the acceleration and deceleration rates via a user variable value, type	**1W(VAR3,130)**
	1VMATH(A,=,VAR3)

AD Deceleration

Syntax	Units	Range of 'n'	Default	See also
aADn	See SCALE	0.01 to 99999.99	10	A AA SCALE VAR VMATH

Description The **AD** command will set or report the programmed linear deceleration rate of the motor. The deceleration value assigned to the **AD** command is overwritten, if previously set.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the deceleration rate of axis 4 to 320 rps^2, type	**4AD320**
To report the current deceleration rate of axis 4, type	**4AD**
The response is	**320*
Overrange value	**AD100027**
Will be reported as...	**E* (meaning error)

To set the acceleration and deceleration rates via a user variable value, type..	**1W(VAR3,130)**
	1VMATH(A,=,VAR3)

D Distance

Syntax	Units	Range of 'n'	Default	See also
aDn	**See SCALE**	**-2,147,483,648 to 2,147,483,647**	**-**	**M SCALE VAR VMATH**

Description The **D** command will set or report the programmed move distance. The value programmed is only used for preset moves. In MC (Move Continuous), the direction is observed. The **D** command can also be set indirectly by using a user variable in combination with the **VMATH** command.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the move distance of axis 2 to 15000 steps type ...	**2D15000**
To report the current programmed move distance of axis 2, type........	**2D**
The controller responds with	**15000*
To set the move distance of axis 2 indirectly to 4000,	**2W(VAR3,4000)**
using the user variable VAR3, type........	**2VMATH(D,=,VAR3)**

Notes If a value entered is out of range *E will be reported and the current value will not be altered.

Distance reports the current direction as influenced by the H command in MI (Mode Incremental) only. For example:

```
1MI       ;mode incremental
1D4000    ;set distance to 4000 steps
1D        ;report distance
*4000     ;value reported
1H-       ;change direction
1D        ;report distance
*-4000    ;value reported
```

IF Test condition

Syntax **aIF(system/user_variable,relation,value or system/user_variable)**

Description The **IF** command compares the specified **system variable** or **user variable** with the specified **value**, **system variable** or **user variable**, using the specified **relation**. If the condition is met the next line of code is executed, otherwise it is skipped.

Refer to the table of system variables that can be used for conditional control.

Valid relations for the comparison are:

=	Equals
<>	Does not equal
>	Greater than
<	Less than

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
2W(VAR1,500)        ; set user variable VAR1 to a value of 500 on Axis 2
2W(VAR2,800)        ; set user variable VAR2 to a value of 800 on Axis 2

2IF(PA,>,450)       ; if system variable PA (absolute position) > 450 steps
2O(1XX)             ;   then set output 1

2IF(PA,>,VAR1)      ; if system variable PA (absolute position) > 500 steps
2O(X1X)             ;   then set output 2

2IF(VAR2,<>,VAR1)   ; if user variable VAR2 is not equal to VAR1
2O(XX1)             ;   then set output 3
```

Using inputs

```
2IF(IN,<>,1X00X)    ; if input does not match the pattern
2O(1X1)             ;   then set outputs 1 & 3
```

Note If you wish to use the **IF** command during motion, command queuing (system variable CQ) must be set for continuous execution (CQ=0).

LOOP Repeat user code

Syntax **aLOOP(label,VARi)**

Description The **LOOP** command repeatedly calls a **labelled** block of code a number of times specified by the value of the **VARi**, the range being 0 to 65000.
Note: If the number of cycles is set to 0 the loop will continue indefinitely.

Nesting of loops up to 5 levels is permitted.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example Execute the predefined program 'TEST' 10 times.

```
1W(VAR0,10)              ; load VAR0 with a value of 10
1LOOP(TEST,VAR0)         ; loop program 'TEST' 10-times
```

Note In the above example, the number of times the program "TEST" is run is taken from the value of VAR0 at the point in time when the LOOP command is executed. If the value of VAR0 is subsequently changed, after the Loop command has started execution, this will not affect the number of times the loop is run.

T(var) Time Delay by Variable

Syntax	Units	Range of 'n'	Default	See also
aT(VARi)	milliseconds	50 to 10000	none	VAR

Description The **T(var)** command pauses program execution for a period of time in milliseconds, set by the value of the user variable **VARi**. Timing resolution is in 50ms increments. Any time value specified within the range 50 to 10,000ms will be rounded down to the nearest 50ms increment. Any value programmed outside of this range will generate an error (**E out of range*).

The receipt of an immediate command whilst executing a time delay causes the delay to end.

Properties Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
1W(VAR0,550)  ; load VAR0 with a value of 550
1T(VAR0)      ; delay program execution by 550 milliseconds
```

TR Wait for trigger

Syntax **aTR(system/user_variable,relation,value or system/user_variable)**

Description

The **TR** command pauses command execution until the trigger condition is met.

The trigger condition is met if the **relation** between **system variable** or **user variable** and the specified **value**, **system variable** or **user variable** is true.

Valid relations for the comparison are:

=	Equals
<>	Does not equal
>	Greater than
<	Less than

Also see system variable Trigger Timeout (**TT**).

Refer to the table of system variables that can be used for conditional control.

Properties

Immediate or buffered, can be used in labelled block, not saved by SV

Example

```
3TR(PA,>,2000)     ; wait for system variable PA (absolute position) to be
                   ; greater than 2000 steps

3TR(IN,=,X11XX)    ; wait for user inputs 2 and 3 to be high

3W(VAR1,500)       ; set user variable 1 to a value of 500
3W(VAR5,5000)      ; set user variable 5 to a value of 5000
3TR(PA,<,VAR5)     ; wait for system variable PA (absolute position) to be
                   ; less than 5000 steps
3TR(VAR1,<,VAR5) ; wait for user variable VAR1 to be less than VAR5
```

Notes

If you wish to use the **TR** command during motion, command queuing (system variable **CQ**) must be set for continuous execution (CQ=0).

Issuing a **K** or **S** from the command line will clear a trigger condition.

If the input command buffer is filled whilst waiting for a trigger *E will be reported (assuming **EX** is set to speak whenever), and the status LED will continually flash red then green. To clear this condition, cycle the power.

V Velocity

Syntax	Units	Range of ‘n’	Default	See also
aVn	**See SCALE**	**0.001 to 5000.000**	**1**	**PROFILE** **SCALE** **VMATH**

Description Velocity command **V** sets or reports the programmed velocity of the motor. The **V** command can also be set indirectly by using a user variable in combination with the **VMATH** command.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the velocity of axis 3 to 25 rps, type	**3V25**
To report the current velocity of axis 3, type	**3V**
The controller responds with..	**25.0*

No units are reported.

To set the velocity of axis 3 indirectly to 10rps, using the user variable VAR3, type .... **3W(VAR3,10)** **3VMATH(V,=,VAR3)**

Note [1] A programmed value of velocity can be overwritten by a PROFILE command once the USE command has been issued, but subsequent values of velocity can be programmed to override the value in use.

Over range value is V5000, this is reported as **E*, value out of range.

VAR Numeric Variable Assignment

Syntax	Units 'i'	Range 'n'	Default	See also
aVARi,n	0 to 7	-2,147,483,647 to 2,147,483,647	-	VMATH SV

Description The user variables have the Mnemonic symbol **VAR**, followed by an identifying number in the range 0 to 7. The variables are all of a long-integer type; no real (float) or binary values are supported. User variables are written to and read from in the same manner as the system variables, using the **W** and **R** commands.

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example

To set the value of VAR0 of axis 1 to 1324, type..........	**1W(VAR0,1324)**
To determine the value of VAR0 of axis 1, type............	**1R(VAR0)**
The response is	**1324*
Overrange value	**1W(VAR0,2222222222)**
Will be reported as	**E* (meaning error)

Note Variable values are only stored if saved with the SV command.

VMATH Maths Command for User Variables

Syntax **aVMATH(destination,operand,source)**

Description Use to perform mathematical operations on system and user variables.

destination is the target variable (user variable or valid system parameter*)
source is the source variable (user variable or valid system parameter*)
operand selects the mathematical operation to be performed by VMATH:

= equate
+ addition
- subtraction
***** multiplication
/ division

*valid system parameter listed in figures 1 and 2

To write a user variable (source) value to a system variable (destination):

1VMATH(System Variable,=,VARi)

To write a system variable (source) value to a user variable (destination):

1VMATH(VARi,=,System Variable)

To multiply a system variable (source) with a user variable (destination):

1VMATH(VARi,*,System Variable)

Properties Immediate or buffered, can be used in labelled block, saved by SV

Example Writing to a ‘System Variable’ from a ‘User Variable’:

Set user variable VAR0 to 4000……………………….. **1W(VAR0,4000)**
Set system variable EW equal to value of VAR0…. **1VMATH(EW,=,VAR0)**
Read back value of system variable EW…………… **1R(EW)**
The response is……………………………………….. ***4000**

Reading from a ‘System Variable’ to a ‘User Variable’:
Set VAR3 equal to value of system variable PC….. **1VMATH(VAR3,=,PC)**
Read back value of VAR3…………………………… **1VAR3**
The response is……………………………………….. ***300**

Note Variable values are only stored if saved with the SV command.

When carrying out VMATH functions, it should be noted that only long integer values are used, in the range ±2147483647. If the result of a VMATH operation gives a non-integer value, the result is truncated not rounded.

Figure 1 details the ViX system variables which can be used in conjunction with the VMATH command. Figure 2 details the EASI-code commands which can be used in conjunction with the VAR and VMATH commands.

System Variable	Description	Source Syntax	Destination Syntax	Destination Range
AI	Analogue Input	1VMATH(VARi,=,AI)	-	±2047
CL	Current Clamp	1VMATH(VARi ,=,CL)	1VMATH(CL,=,VARi)	1 to 100%
CR	Current Reference	1VMATH(VARi ,=,CR)	-	counts
EW	Error Window	1VMATH(VARi,=,EW)	1VMATH(EW,=,VARi)	0 to 65535
PA	Position Actual	1VMATH(VARi,=,PA)	-	± 2147483648
PC	Peak Current	1VMATH(VARi,=,PC)	1VMATH(PC,=,VARi)	100 to 400%
PE	Position Error	1VMATH(VARi ,=,PE)	-	± 65535
PF	Position Following	1VMATH(VARi ,=,PF)	-	± 2147483648
PI	Position Incremental	1VMATH(VARi ,=,PI)	-	± 2147483648
PM	Position Master	1VMATH(VARi,=,PM)	-	± 2147483648
PR	Position Registration	1VMATH(VARi ,=,PR)	-	± 2147483648
PS	Position Secondary	1VMATH(VARi,=,PS)	-	± 2147483648
PT	Position Target	1VMATH(VARi,=,PT)	-	± 2147483648
TL	Tracking Limit	1VMATH(VARi,=,TL)	1VMATH(TL,=,VARi)	0 to 65535
TT	Trigger Timeout	1VMATH(VARi,=,TT)	1VMATH(TT,=,VARi)	0 to 65*
AB	Analogue Deadband	1VMATH(VARi,=,AB)	1VMATH(AB,=,VARi)	0 to 255
AO	Analogue Offset	1VMATH(VARi,=,AO)	1VMATH(AO,=,VARi)	±2047
IT	In Position Time	1VMATH(VARi,=,IT)	1VMATH(IT,=,VARi)	1 to 500ms
IW	Integral Window	1VMATH(VARi,=,IW)	1VMATH(IW,=,VARi)	0 to 65535

Figure 1: System Variables *Limited Range when using User Variables

Command	Source Syntax	Destination Syntax		Range	n	
IF	-	1IF(VARi,n,c)	VAR 0-7		Operand >,<,=,<>	VAR 0-7 or Number
TR	-	1TR(VARi,n,c)	VAR 0-7		Operand >,<,=,<>	VAR 0-7 or Number
LOOP	-	1LOOP(label,VARi)	Label	0 to 65000	VAR 0-7	
T	-	1T(VARi)	VAR 0-7	50-10,000	50ms – 10s	
A	1VMATH(VARi,=,A)	1VMATH(A,=,VARi)	A	1 to 99999*	Operand =	VAR 0-7
AA	1VMATH(VARi,=,AA)	1VMATH(AA,=,VARi)	AA	1 to 99999*	Operand =	VAR 0-7
AD	1VMATH(VARi,=,AD)	1VMATH(AD,=,VARi)	AD	1 to 99999*	Operand =	VAR 0-7
D	1VMATH(VARi,=,D)	1VMATH(D,=,VARi)	D	±2147483647	Operand =	VAR 0-7
V	1VMATH(VARi,=,V)	1VMATH(V,=,VARi)	V	1 to 5000*	Operand =	VAR 0-7

Figure 2: Commands *Limited Precision when using User Variables