MUMPS Standard error codes

M1, Naked indicator undefined

Introduced in the 1995 ANSI M[UMPS] language standard.

The naked indicator is undefined at the start of a M[UMPS] process, or after a reference to an unsubscripted global variable has been made.
The sequence of commands `SET ^A=3,^(4)=5` would cause this error to occur.

M2, Invalid combination with P fncodatom

Introduced in the 1995 ANSI M[UMPS] language standard.

An fncodatom is one of the special formatting parameters of the function **$FNUMBER**. Currently, the valid codes that may be specified as such are `+` (plus, force plus sign), `,` (comma, insert delimiters every three digits), `-` (minus, suppress minus sign), `p` or `P` (represent negative numbers in parentheses) and `t` or `T` (trailing sign). Any incompatible combination of these codes or any other character specified in the second parameter of the function **$FNUMBER** will cause this error to occur.

```
WRITE $FNUMBER(AMOUNT,"+Q")
WRITE $FNUMBER(AMOUNT,"-t",3)
WRITE $FNUMBER(AMOUNT,"Z",2)
```

M3, $RANDOM seed less than 1

Introduced in the 1995 ANSI M[UMPS] language standard.

`$RANDOM(6)` returns one of the values 0, 1, 2, 3, 4 or 5.
`$RANDOM(-6)` does not return one of the values -5, -4, -3, -2, -1 or 0, but causes this error to occur.

M4, No true condition in $SELECT

Introduced in the 1995 ANSI M[UMPS] language standard.

This error is most likely to occur in constructions like
`WRITE $SELECT(X>50:"Large",X<50:"Small")`
(Note: in the case that `X=50`).

M5, lineref less than zero

Introduced in the 1995 ANSI M[UMPS] language standard.

This error means that, in the context of a reference to the function **$TEXT**, a reference was encountered that has a negative offset value.

```
WRITE $TEXT(+LINECOUNT^MYROUT)
```
(and the value of local variable `LINECOUNT` is negative).

A reference a **DO**, **GOTO** or **JOB** command with a negative offset will cause an error with code M12.

M6, Undefined lvn

Introduced in the 1995 ANSI M[UMPS] language standard.

This error means that an attempt was made to obtain the value of a local variable, while that local variable has no defined value. Note that a variable that has no defined value may have descendants that do have a defined value.

```
KILL X
SET X(12,13,14)=123
SET:$DATA(X(12)) ABC=X(12)
```

M7, Undefined gvn

Introduced in the 1995 ANSI M[UMPS] language standard.

This error means that an attempt was made to obtain the value of a global variable, while that global variable has no defined value. Note that a variable that has no defined value may have descendants that do have a defined value.

```
KILL ^X
SET ^X(12,13,14)=123
SET:$DATA(^X(12)) ABC=^X(12)
```

M8, Undefined svn

Introduced in the 1995 ANSI M[UMPS] language standard.

The intrinsic special variables (**$X**, **$IO**, etcetera) always have a value. This error means that a reference was encountered to an entity that has the syntax of an intrinsic special variable name, but that does not exist as such, e.g. **$DEVICE** in an implentation that meets the 1990 standard, but not the 1995 standard, or $CPU.

M9, Divide by zero

Introduced in the 1995 ANSI M[UMPS] language standard.

This error means that during the evaluation of an expression containing a division (either common (/) or integer division (\)), a 'righthand' operand was encountered that has a zero value.

```
SET (TOTAL,NUMBER)=0
SET I="" FOR SET I=$ORDER(A(I)) QUIT:I="" DO
 . SET N=N+1,TOTAL=TOTAL+A(I)
 . QUIT
SET AVERAGE=TOTAL/NUMBER
```

This error will occur in the case that array A has no members.

```
SET X="Apples"
SET Y=TOTAL\X+Y
```

This error will also occur for:

```
SET X=N#0
SET X=0**-1
```

M10, Invalid pattern match range

Introduced in the 1995 ANSI M[UMPS] language standard.

If, in a pattern specification, the number of occurrences of a part of a pattern is specified (number.number), the first number gives the lower bound, and the second number the upper bound. If the upper bound is less than the lower bound this error condition occurs.

```
IF X?1.5A3N6.4A
```

M11, No parameters passed

Introduced in the 1995 ANSI M[UMPS] language standard.

When execution 'falls through' and reaches a line that starts with a label that has a parameter list, this error will occur. Labels with a parameter list should only be executed when invoked as a subroutine, or as an extrinsic function or extrinsic variable.

M12, Invalid lineref (negative offset)

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **DO**, **GOTO** or **JOB** command is executed, and the line that is specified is in the form `LABEL+OFFSET^ROUTINE`, this error will occur if the `OFFSET` has a negative value.

A reference to the function **$TEXT** with a negative `OFFSET` will cause an error with code M5.

M13, Invalid lineref (label not found)

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **DO**, **GOTO** or **JOB** command is executed, or when a reference to the function **$TEXT** is evaluated, and the line that is specified is in the form `LABEL+OFFSET^ROUTINE` or `LABEL+OFFSET`, this error will occur if either there is no label in the specified routine that corresponds to the specified label, or the value of `OFFSET` is so high that the reference no longer evaluates to a valid line in the specified routine.

M14, line *level* not 1

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **DO**, **GOTO** or **JOB** command is executed, and the line that is specified is in the form `LABEL+OFFSET^ROUTINE` or `LABEL+OFFSET`, this error will occur if the line that is specified does not have a *level* of 1, i.e. the line starts with one or more period characters. Lines that have a *level* higher than 1 can only be invoked by an argumentless **DO** command.

The *level* of a line is equal to 'number of dots at the start of the line' plus 1.

M15, Undefined index variable

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **FOR** command is executed, and the range specification is of the form FROM:STEP or FROM:STEP:UNTIL, and it is not possible to assign a value to the index variable of the **FOR** loop, this error will occur.

M16, Argumented QUIT not allowed

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **QUIT** command is executed as the termination of an extrinsic function, the value specified in the argument of the **QUIT** command is the value that is returned to the environment that invoked the extrinsic function. When a **QUIT** command is executed in any other context, and an argument is present, this error will occur.

M17, Argumented QUIT required

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **QUIT** command is executed as the termination of an extrinsic function, the value specified in the argument of the **QUIT** command is the value that is returned to the environment that invoked the extrinsic function. When a **QUIT** command is executed in this context, and no argument is present, this error will occur.

M18, Fixed length READ not greater than zero

Introduced in the 1995 ANSI M[UMPS] language standard.

When a **READ** command is executed, and an argument is of the form VARIABLE#LENGTH, this error will occur if the value of LENGTH is less than 1.

M19, Cannot copy a tree or subtree into itself

Introduced in the 1995 ANSI M[UMPS] language standard.

When a MERGE command is executed, and there is an overlap between the source and the destination of the information to be copied, this error will occur.

```
MERGE ^ABC(1,2,3)=^ABC(1,2,3,4,5,6)
MERGE ^ABC(1,2,3,4,5)=^ABC(1,2,3,4)
MERGE ^ABC(1,2,3)=^ABC(1,2,3)
MERGE ^|HERE|ABC(1,2,3)=^|THERE|ABC(1,2,3)
```

(in the last example, the error will occur if the value of HERE happens to be equal to the value of THERE).

M20, line must have formallist

Introduced in the 1995 ANSI M[UMPS] language standard.

When an extrinsic function is invoked (e.g. $$XXX^YYY(P1,P2)), and the label in the routine that is invoked to calculate the function value does not have a parameter list (in this example, the label XXX in routine YYY), this error will occur.

M21, Algorithm specification invalid

Introduced in the 1995 ANSI M[UMPS] language standard.

When a label occurs with a parameter list (a formallist), and the same name occurs multiple times in this parameter list, this error will occur.

Note that it does make sense to be able to have the same name multiple times in a parameter list when a function or subroutine is invoked (actuallist), but not in the formal parameter list at the start of that function or subroutine.

```
SET SUM=$$ADD(A,A)
```

is a valid function call.

```
ADD(X,X) QUIT X+X
```

will cause this error to occur; most likely, this should have been

```
ADD(X,Y) QUIT X+Y
```

M22, SET or KILL to ^$GLOBAL when data in global

Introduced in the 1995 ANSI M[UMPS] language standard.

The structured system variable **^$GLOBAL** contains information about global variables (allocation, protection, access control, etcetera). Certain information about global variables can only be entered into these structured variables before the global variables that they describe are actually created. When creating the global variable in question, the parameters specified in the structured system variable will be applied, and it may not generally be possible to undo this without completely re-creating the global variable in question.

An attempt to define or delete such information in this structured system variable about a global variable that currently exists will cause this error to occur.

M23, SET or KILL to ^$JOB for non-existent job number

Introduced in the 1995 ANSI M[UMPS] language standard.

The structured system variable **^$JOB** contains information about M[UMPS] processes (execution status, ownership, etcetera). Information about M[UMPS] processes can only be defined or modified in these structured variables while the process in question exists in the system.

An attempt to define or delete information in this structured system variable about a M[UMPS] process that currently does not exist will cause this error to occur.

M24, Change to collation algorithm while subscripted local variables defined

Introduced in the 1995 ANSI M[UMPS] language standard.

The structured system variable **^$GLOBAL** contains information about global variables (allocation, protection, access control, etcetera). Certain information about global variables can only be entered into these structured variables before the global variables that they describe are actually created. When creating the subtree in question, the collation algorithm specified in the structured system variable will be applied, and it may not generally be possible to undo this without completely re-creating the global variable in question.

The 1995 ANSI M[UMPS] language standard allows for a specification of a collation algorithm at the 'top level' of a global variable only.

An attempt to define or delete information related to the collation algorithm of a subtree in a global variable in this structured system variable while that subtree exists will cause this error to occur.

M25, Attempt to modify currently executing routine

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error code means that an attempt was made to execute an **RSAVE** command specifying the currently executing routine as the routine to be re-created. When this error occurs, the routine in question will not be modified.

M26, Non-existent environment

Introduced in the 1995 ANSI M[UMPS] language standard.

When a reference is evaluated that contains an environment specification, and the specified environment does not currently exist, this error will occur.

Note that this error will not occur while building a definition in a structured system variable.

Networks may go down
```
SET X=^|"LONDON"|CUSTOMER(ID)
```

Or an environment may not exist at all...
```
SET X=^|"ATLANTIS"|CUSTOMER(ID)
DO ^|"ATLANTIS"|GETINFO
```

M27, Attempt to rollback a transaction that is not restartable

Introduced in the 1995 ANSI M[UMPS] language standard.

One of the activities that may occur in the context of transaction processing is a *restart*. A restart is a second (or third, or...) attempt to process the complete *transaction*. Whether or not a *transaction* is actually restarted, depends on whether it is restartable. A *transaction* is restartable if the initiating **TSTART** command specifies a restartargument. A *restart* of a restartable *transaction* causes execution to resume with the initial **TSTART** command. A *restart* of a non- restartable *transaction* ends the *transaction* and causes this error to occur.

```
TSTART (A,B,C):TRANSACTIONID="Example"
; ...
; set up transaction
;...
; Programmer can initiate restart
IF PROBLEMS TRESTART
; ...
; Complete the transaction
TCOMMIT
```

M28, Mathematical function, parameter out of range

Introduced in the 1995 ANSI M[UMPS] language standard.

While the 1995 ANSI M[UMPS] language standard introduces the concept of external calls and external libraries, that standard does not yet define any specific external callable routines.

A future ANSI M[UMPS] language standard will contain a set of mathematical functions that are accessible through the external call syntax.

When one of these functions is called with one or more inappopriate parameters, this error will occur.

```
SET X=$%SQRT^MATH(-1)
SET X=$%ARCSIN^MATH(5.3)
```

M29, SET or KILL on ssvn not allowed by implementation

Introduced in the 1995 ANSI M[UMPS] language standard.

While the definition of the M[UMPS] language may allow for certain syntactical constructs, not all implementation will allow the use of these at all times.

When an attempt is made to (re-)define or remove information in a structured system variable, while the implementation does not allow such activity, this error will occur.

```
KILL ^$JOB(JOBID)
KILL ^$ROUTINE("%H")
SET ^$GLOBAL("^DD","CHARACTER")="Old-Norse"
```

M30, Reference to glvn with different collating sequence within a collating algorithm

Introduced in the 1995 ANSI M[UMPS] language standard.

In `^$CHARACTER(setname,"COLLATE")`, an algorithm may be specified that will be used internally by the M[UMPS] system to establish the collating order of for a character-set. The string that will be used internally for collating purposes is computed by a function call equivalent to `SET internal=$$entry^routine(actualvalue)`. If such a function is invoked, and a reference is evaluated to a variable that has a different collating algorithm than the one that is being computed, this error will occur.

M31, controlmnemonic used for device without a mnemonicspace selected

Introduced in the 1995 ANSI M[UMPS] language standard.

Usage of the special device control functions (controlmnemonic) that are available through the `WRITE /function` syntax can only take place after a function-library has been selected in a **USE** command or in an **OPEN** command. If such a function reference occurs, while no library (mnemonicspace) is activated, this error will occur.

M32, controlmnemonic used in user-defined mnemonicspace which has no associated line

Introduced in the 1995 ANSI M[UMPS] language standard.

M33, SET or KILL to ^$ROUTINE when routine exists

Introduced in the 1995 ANSI M[UMPS] language standard.

The structured system variable **^$ROUTINE** contains information about routines (allocation, protection, access control, etcetera). Certain information about routines can only be entered into

these structured variables before the routines that they describe are actually created. When creating the routine in question, the parameters specified in the structured system variable will be applied, and it may not generally be possible to undo this without completely re-creating the routine in question.

An attempt to define or delete such information in this structured system variable about a routine that currently exists will cause this error to occur.

M34, --- currently unassigned ---

M35, Device does not support mnemonicspace

Introduced in the 1995 ANSI M[UMPS] language standard.

One of the parameters that can be specified on **OPEN** and **USE** commands is a list of names of mnemonicspaces. A mnemonicspace may be compared to a function library: mnemonicspaces contain functions (called controlmnemonics) that may be applied to the device in question.

When a mnemonicspace is specified in an **OPEN** or **USE** command, and that mnemonicspace is not supported by the device to which it is being applied, this error will occur.

M36, Incompatible mnemonicspaces

Introduced in the 1995 ANSI M[UMPS] language standard.

One of the parameters that can be specified on **OPEN** and **USE** commands is a list of names of mnemonicspaces. A mnemonicspace may be compared to a function library: mnemonicspaces contain functions (called controlmnemonics) that may be applied to the device in question.

When a combination of mnemonicspaces is specified in an **OPEN** or **USE** command, and one of these mnemonicspaces is not compatible with one or more of the other ones, this error will occur.

M37, READ from device identified by the empty string

Introduced in the 1995 ANSI M[UMPS] language standard.

When the current device is closed, the value of **$IO** will be set to the empty string. The 1995 ANSI M[UMPS] language standard does not allow read operations to take place while the value of **$IO** is empty. If, under these circumstances, a **READ** command is executed, this error will occur.

Note: there is a proposal (March 1995: SC#12 Type B) that would allow read operations from a "null device".

M38, Invalid ssvn subscript

Introduced in the 1995 ANSI M[UMPS] language standard.

The subscripts of local and global variables may be any string. There are some restrictions in that most implementations do not allow control characters in subscripts, and all implementations observe a maximum length for subscripts, but those are the only 'normal' limitations.
For structured system variables, however, the subscripts are bound by semantical constraints: the first subscript in **^$ROUTINE** must be the name of a routine, for instance. When a reference is made to s structured system variable, and one of the subscripts does not meet its semantical constraints, this error will occur.

Approved for addition in a future ANSI M[UMPS] Windowing Application Programmer's Interface standard.

This error will occur when an attempt is made to register an event handler (`SET ^$JOB(process,"EVENT",eventclass,eventid)=entrypoint`) and the value of either `class` or `id` is not a valid one.

M39, Invalid $NAME argument

Introduced in the 1995 ANSI M[UMPS] language standard.

When a reference to the function **$NAME** is evaluated, and the second parameter of the function **$NAME** has a negative value, this error will occur.

```
SET X=$NAME(^(2),SUB)
```
while the value of `SUB` is less than 0.

M40, Call-by-reference in JOB actual

Introduced in the 1995 ANSI M[UMPS] language standard.

The **JOB** command may be invoked to start at a specified label, with a parameterlist that specifies the initial values of certain variables in the newly started job. When one or more of the parameters specified in the actual parameter list when the **JOB** command is executed is passed by reference (preceded by a dot), this error will occur.

M41, Invalid LOCK argument within a TRANSACTION

Introduced in the 1995 ANSI M[UMPS] language standard.

In the context of transaction processing, it is not allowed to release resources within a *transaction*, if those resources were obtained outside of the *transaction* in question. This restriction is necessary to allow *transactions* to be restartable.

When, within a *transaction*, a **LOCK** command is executed that releases a name from the LOCK list that was not entered into the LOCK list within the context of the current *transaction*, this error will occur.

M42, Invalid QUIT within a TRANSACTION

Introduced in the 1995 ANSI M[UMPS] language standard.

Transactions are delimited by a **TSTART** and **TCOMMIT** commands. A **TCOMMIT** command must occur at the same or a deeper **DO** level than the corresponding **TSTART** command.

When a **TSTART** command has been executed, and a **QUIT** command is executed that returns from the **DO** level at which the **TSTART** command was encountered, this error will occur.

```
DO INIT,PROCESS,EXIT
QUIT
;
INIT ... TSTART ... QUIT
PROCESS ... QUIT
EXIT ... TCOMMIT ... QUIT
```

The above processing model is not compatible with transaction processing. The complete definition of a *transaction* must be in one **DO**-able module.

M43, Invalid range ($X, $Y)

Introduced in the 1995 ANSI M[UMPS] language standard.

One addition to the 1995 ANSI M[UMPS] language standard is that it is now allowed to **SET $X**=value and **SET $Y**=value.

When the value that is assigned to **$X** or **$Y** in this context is not a valid value for the variable in question, this error will occur.

M44, Invalid command outside of a TRANSACTION

Introduced in the 1995 ANSI M[UMPS] language standard.

Transactions are delimited by a **TSTART** and **TCOMMIT** commands. A trancaction may be cancelled by means of a **TROLLBACK** command, or restarted by means of a **TRESTART** command.

When a **TCOMMIT**, **TRESTART** or **TROLLBACK** command is encountered while no transaction is active (no **TSTART** command executed, or **$TLEVEL**=0), this error will occur.

M45, Invalid GOTO reference

Introduced in the 1995 ANSI M[UMPS] language standard.

One addition in the 1990 ANSI M[UMPS] language standard is that it has become possible to define blocks of code that can only be entered at one point by means of an argumentless **DO** command. Within such blocks, deeper nested blocks may be defined. Within one such block, control may be transferred using a **GOTO** command.

When a **GOTO** command is executed, and the destination of the executed argument is a line within an 'indented' block, and the **GOTO** command does not originate from a line at the same level of indentation in the same 'indented' block, this error will occur.

Note: in his article ***GOTO Command Considered Harmful*** (March 1968, Communications of the ACM, Volume 11/3), Edsger W. Dijkstra warns against a certain improper usage of **GOTO** commands. This restriction in the language helps to prevent such improper usage.

M46, Invalid attribute name

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

If an application attempts to assign a value to a reserved attribute name, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?1.

M47, Invalid attribute value

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

If a value domain is specified for an attribute, all values outside the specified domain are reserved. Unless specifically stated otherwise, if an application attempts to assign an attribute value outside of the specified domain of that attribute, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

Assume that an application has defined two windows, A and B. If, for window A, the application attempts to assign a value to the PARENT attribute, naming window B as the parent, and the DISPLAY attributes for windows A and B are different, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

If an application attempts to assign a value of `"TRUE"` to a TIED attribute, and the window's parent is a display, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?2.

M48, Nonexistent window, element or choice

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

Values may be assigned to attributes by means of **SET** and **MERGE** commands. If an application attempts to assign a value in this way to an attribute of a window or element, while that window or element does not exist at the time that the command in question is executed, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?3.

M49, Invalid attempt to set focus

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

If an application attempts to set focus to a window or gadget that is not currently capable of receiving focus, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

There are many reasons why a window or gadget may be unable to receive focus: it may be invisible, not active, disabled by the value of its MODAL attribute, or reside on a different display.

If an application attempts to assign a value to a NEXTG attribute that does not identify a gadget that has been defined for the window in question, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?4.

M50, Attempt to reference a non M-Term window in an OPEN command

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

An MTERM window is opened by specifying the `"MTERM"` mnemonic-space in an **OPEN** command. If the device that is being OPENed does not yet exist as a window in the structured system variable **^$WINDOW**, a window is created in **^$WINDOW**, with its window name equal to the name of the device specified in the **OPEN** command, and its TYPE attribute equal to `"MTERM"`.
If the device specified in the **OPEN** command already exists in **^$WINDOW**, and its TYPE attribute is not equal to `"MTERM"`, this error will occur. For this error, there is no corresponding value in special variable **$EREF**.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?5.

M51, Attempt to destroy M-Term window prior to CLOSE

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

If the device specified in a **CLOSE** command designates an MTERM window that was created by the execution of an **OPEN** command, the window in question will be destroyed.
If the device specified in a **CLOSE** command designates an MTERM window that was created by the assignment of values to attributes in structured system variable **^$WINDOW**, the **CLOSE** command will not cause the window to be destroyed, and this error will occur. For this error, there is no corresponding value in special variable **$EREF**.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?6.

M52, Required attribute missing

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

If an application attempts to assign a value to an attribute while not all criteria to be allowed to do so are met, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

For a DEFBUTTON attribute the criteria that need to be met are:

- the gadget must be a 'child' of the window;
- the TYPE attribute of the gadget must be `"BUTTON"`.

For gadgets and for menus not displayed as menu bars or as submenus the criteria that need to be met are:

- a value must be assigned to the POS attribute when the gadget is created,
- both a horizontal and a vertical position must be defined.

For a symbol gadget the criterion that needs to be met is:

- the RESOURCE attribute must have a value.

For a generic box or a group frame gadget the criterion that needs to be met is:

- a value must be assigned to the SIZE element attribute when the gadget is created.

For all gadgets a criterion that needs to be met is:

- a value must be assigned to the TYPE element attribute when the gadget is created.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?7.

M53, Invalid argument for font function

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

Each of the functions **$WFONT**, **$WTFIT** and **$WTWIDTH** has a parameter that specifies the units in which some of the other parameters are expressed. The unit- specification parameter must be equal to either `"PIXEL"` or `"POINT"`. If any other value is specified for the unit-speficication parameter, this error will occur.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?8.

M54, Attempt to create non-modal child of a modal parent

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

If an application attempts to create a child window of a modal window which is not also a modal window, this error will occur. After this, special variable **$EREF** will contain a value that indicates the reference to the structured system variable for which the error occurred.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?9.

M55, Invalid nested ESTART command

Introduced in the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface.

During call back processing for events and prior to execution of a nested **ESTART** command, the FOCUS display attribute for the display on which the window appears that is associated with the event being processed, must be assigned a value which identifies a window that has its MODAL window attribute equal to `"APPLICATION"`. If the FOCUS attribute is not assigned such a value, this error will occur.

In earlier printed versions of the 1995 ANSI M[UMPS] Windowing Application Programmers' Interface, this error was identified as M?10.

M56, Name length exceeds implementation's limit

Approved for inclusion in a future ANSI M[UMPS] language standard.

In the 1977 ANSI M[UMPS] language standard, the length of a name (for routines, as well as local and global variables) was restricted to 8 characters. Implementations were not required to interpret any characters beyond the 8th while determining the unique meaning of a name.
in a future ANSI M[UMPS] language standard, the length of a name is no longer restricted. For portability, a maximum of 31 characters is specified, but implementations may allow for longer names. If an application uses a name that is longer than the number of characters that a specific implementation allows (i.e. more than 31 characters), this error will occur.

The purpose of this error is to warn about possible portability problems: if an application has taken advantage of an implementation that offers names longer than the portability limit, and subsequently is run on an application that binds names to a lower limit (either the portability limit, or a number between the portability limit and the 'roomier' implementation's limit), this error will occur, rather than that different variables are mapped onto the same internal unique identifier.

M57, More than one defining occurrence of label in routine

Introduced in the 1995 ANSI M[UMPS] language standard.

While the standard defines that for each label there may be at most one occurrence in a routine where the label in question occurs at the start of a line, the standard does not define when this specific error will occur.
Implementations that issue this error at all, will typically report this error when the routine in question is created.

M58, Too few formal parameters

Introduced in the 1995 ANSI M[UMPS] language standard.

When a parameterized entry is called with less actual parameters than formal parameters, the 'missing' parameters will be treated as having a **$DATA** attribute of 0 within the invoked procedure. When a parameterized entry is called with mote actual parameters than formal parameters, this error will occur.

M59, Environment reference not permitted for this ssvn

Approved for inclusion in a future ANSI M[UMPS] language standard.

SETting, **KILL**ing and referencing "nodes" in structured system variables is only allowed when the semantics for the operation in question are explicitly defined. Any reference to a ssvn, where an environment is specified, and no semantics are defined for the meaning of an environment in that context, will cause this error to occur.

M60, Undefined ssvn

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error means that an attempt was made to obtain the value of a structured system variable, while that variable has no defined value. Note that a variable that has no defined value may have descendants that do have a defined value.

```
SET X=^$ROUTINE
SET X=^$ROUTINE(somename)
```

M61, Attempt to OPEN file with conflicting ACCESS parameters

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M62, Illegal value for ACCESS parameter while attempting to OPEN file

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M63, Illegal value for DISPOSITION parameter while attempting to CLOSE file

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M64, Illegal value for RENAME parameter while attempting to CLOSE file

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M65, Illegal value for VOLUME label

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M66, Illegal value for DENSITY parameter

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M67, Illegal value for ACCESS parameter

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M68, Illegal value for MOUNT parameter

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M69, Attempted tape I/O while no tape mounted

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M70, Illegal value for BLOCKSIZE parameter

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M71, Attempt to read data block larger than buffer size

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M72, Illegal value for recordsize parameter

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M73, Invalid usage of devicekeyword NEWFILE

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M74, Illegal value for TRANSLATION parameter

The proposal to introduce this error code was rescinded in June 1995.
While the error code remains reserved, this particular error will never occur.

M75, String length exceeds implementation's limit

Approved for inclusion in a future ANSI M[UMPS] language standard.

For portability, the length of strings is limited. The 1977 ANSI M[UMPS] language standard limited this length to 255 characters, in a future ANSI M[UMPS] language standard, this limit will be raised to:

- 510 characters for namevalues
- 32,767 characters for values in local variables
- 510 characters for values in global variables
- 32,767 characters for values in structured system variables

Implementations may offer longer strings than the portability limit.

If an implementation attempts to create a string that is longer than the string length limit for the implementation in question, this error will occur.

This error will not occur for any string that has a length greater than the portability limit, but less that the limit of the implementation.

```
SET X="",$ETRAP="GOTO ERROR"
 WRITE !,"This should work with the 1977 standard"
 FOR I=1:1:255 SET X=X_$CHAR(I#26+65)
 DO SHOW
 WRITE !,"The next code-line is not portable with"
 WRITE !,"the 1990 standard, but is"
 WRITE !,"portable with the 1995 standard."
 FOR I=256:1:510 SET X=X_$CHAR(I#26+65)
 DO SHOW
 WRITE !,"The next code-line is not portable with"
 WRITE !,"the 1995 standard, but is"
 WRITE !,"portable with the proposed millennium standard."
 FOR I=511:1:32767 SET X=X_$CHAR(I#26+65)
 DO SHOW
```

```
 WRITE !,"The following remains non-portable:"
 FOR I=1:1 SET X=X_$CHAR(I#26+65)
 ; The above line will cause error M75 to occur
 ;
SHOW WRITE !,"Now ",$LENGTH(X)," characters."
 QUIT
 ;
ERROR DO SHOW SET $ETRAP="" QUIT
```

M76, TCP socket state incorrect for CONNECT or LISTEN

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M77, TCP deviceattribute missing

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~ A future version of this booklet will contain more information about this error condition. ~~]

M78, TCP devicekeyword missing

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M79, TCP socket allocated to another device

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M80, Network error not otherwise specified

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M81, Unable to establish network connection

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M82, Network connection suspended: wait to resume

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M83, Network connection lost

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M84, Network protocol error: invalid client message

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M85, Network protocol error: invalid server message

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M86, Cannot relinquish device with I/O pending

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M87, Network buffer overflow

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M88, Non-existent routine

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error means that an attempt was made to execute an **RLOAD** command which tried to copy a routine that does not exist into a local or global variable.

M89, Specified pattern is not a subpattern

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M90, Invalid namevalue

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when an attempt is made to replace the value of a subscript (**SET $QSUBSCRIPT**(name,sequence)=new), and the namevalue that is specified in the reference to the intrinsic function **$QSUBSCRIPT** is not a valid one.

M91, Routine source is not available

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M92, Mathematical overflow

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when the result of a mathematical operation yields a result that is too large (in absolute value) to be represented as a number. Each of

```
SET N=1 FOR  SET N=N*2
SET N=-1 FOR  SET N=N*2
```

will eventually produce this error.

On the other hand

```
SET N=1 FOR  SET N=N/2 QUIT:'N
```

will eventually terminate when the value of N is small enough that the next division by two will yield a zero value.

M93, Mathematical underflow

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error code was reserved for the situation when the result of a mathematical operation yields a result that is too small (in absolute value) to be represented as a number.

Later it was decided to follow the practice of allowing such results to be represented as a zero value.

```
SET N=1 FOR  SET N=N/2 QUIT:'N
```

will eventually terminate when the value of N is small enough that the next division by two will yield a zero value.

M94, Attempt to compute zero to the zeroth power

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when both operands of the exponentiation operator have a value that evaluates to 0 (zero).

```
"Five Apples"**"Three Pears"
```

will cause this error, while

```
"5 Apples"**"3 Pears" = 125
```

M95, Exponentiation returns complex number with non-zero imaginary part

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when the result of an exponentiation operation is a complex number with a non-zero imaginary part. This may occur when a negative number is raised to a fractional power.

Reference	Value
`-1**3`	`-1`
`-1**4`	`1`
`-32**0.2`	`-2`
`-1**.5`	Error M95
`-8**(1/3)`	Error M95 (1/3 is never exactly the right value to make this result be `-2`

M96, Attempt to assign value to already valued write-once ssvn

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M97, Routine associated with user-defined ssvn does not exist

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M98, Resource unavailable

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M99, Invalid operation for context

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~ A future version of this booklet will contain more information about this error condition. ~~]

M100, Output time-out expired

Approved for inclusion in a future ANSI M[UMPS] language standard.

The ability to specify an "output time-out" is added. An output time-out is specified as a device parameter in an OPEN or USE command. Examples are shown with the USE command.

When an output time-out occurs, i.e. the M[UMPS] system is not able to process output to a specific device within an established time-frame, this error will occur.

Until this error is cleared through external action, the value of `^$D[EVICE](device,"OUTSTALLED")` will be 1 (true).

M101, Attempt to assign incorrect value to $ECODE

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~~~ A future version of this booklet will contain more information about this error condition. ~~~]

M102, Simultaneous synchronous and asynchronous event class

Approved for inclusion in a future ANSI M[UMPS] Windowing Application Programmer's Interface standard.

This error will occur when an attempt is made to register an error handler (**`SET ^$JOB`**`(process,"EVENT",eventclass,eventid)=entrypoint`) and the event is already registered for that process, using the opposite type of processing (i.e. attempting asynchronous processing while synchronous processing is already enabled, or the other way about).

M103, Invalid event identifier

Approved for inclusion in a future ANSI M[UMPS] Windowing Application Programmer's Interface standard.

This error occurs when an attempt is made to trigger an event (see **ETRIGGER** command), and the value of the event-ID is not a valid one.

Which values are valid for event-IDs depends on the event-class, and may be implementation-

specific.

M104, IPC event identifier is not a valid job-number

Approved for inclusion in a future ANSI M[UMPS] Windowing Application Programmer's Interface standard.

This error occurs when an attempt is made to trigger an event for another process (see the **ETRIGGER** command), and the value of the process-identifier is not a valid job-number.

M105, Object not currently accessible

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when a reference is made to an object and the referenced instance of that object is not currently accessible.

A frequent cause for this to happen is that an external process deleted the object.

M106, Object does not support requested method or property

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when a reference is made to a method or property, and the referenced object doed not have such a method or property.

M107, Object has no default value

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when a reference to an object is made in a context where a value of data type MVAL is expected. When the object in question has a default property, the value of that property will be returned, otherwise this error will occur.

M108, Value is not of data type OREF

Approved for inclusion in a future ANSI M[UMPS] language standard.

This error will occur when a reference is made to a value that is expected to be of data type OREF, while the actual value has a different data type.

M109, Undefined devicekeyword

Approved for inclusion in a future ANSI M[UMPS] language standard.

When, in the context of an OPEN or USE command, a parameter is used that includes a devicekeyword that is not supported for the device and mnemonicspace in question, the implementation may or may not cause an error to occur. If the implementation forces an error to occur, this will be the error code that will be returned.

M110, Event identifier not available

Approved for inclusion in a future ANSI M[UMPS] Windowing Application Programmer's Interface standard.

This error will occur when an attempt is made to register an event handler (`SET ^$JOB(process,"EVENT",eventclass,eventid)=entrypoint`) and there are no system-resources available to successfully register the event handler in question.

M111, Invalid number of days for date

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~ A future version of this booklet will contain more information about this error condition. ~]

M112, Invalid number of seconds for time

Approved for inclusion in a future ANSI M[UMPS] language standard.

[~ A future version of this booklet will contain more information about this error condition. ~]
