UNI/O Serial EEPROM Driver

The Microchip 11XX series of EEPROMs are perfectly suited to recording small amounts of data in a Micromite based project. They are available in a range of capacities up to 2 Kbytes and come in an easy to use TO-92 package (a thru hole package that looks like a standard transistor) with only one pin used for communicating with the Micromite.

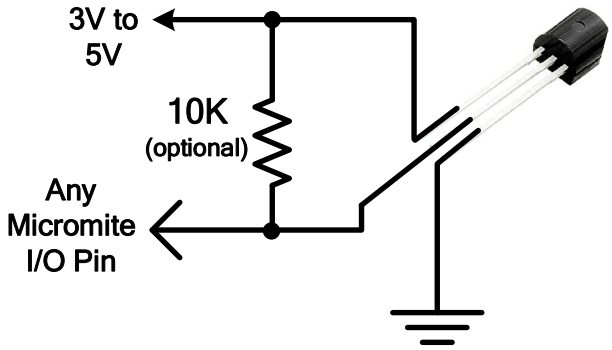
They draw very little current (1µA when not being accessed) and any memory location can be rewritten up to a million times. This last characteristic means that it is possible to sequentially log a value (for example, a temperature) once a second and the EEPROM would not exceed its endurance specification even after many years.

The 11XX series is also low cost, about 20 to 30 cents for a single unit direct from Microchip.

Typical part numbers that work well with the Micromite are the **11AA160-I/TO** (16Kbit TO-92 package) or the **11AA160T-I/TT** (16Kbit SOT-23 package). For more details visit the Microchip website: http://www.microchip.com/wwwproducts/en/11AA160

Connections

The 11XX EEPROM will work off 3.3V or 5V and can be directly connected to any I/O pin on the Micromite. A pullup resistor can be used to reduce noise and errors on the signal line but normally is not required. Note that if the device is powered from 5V then a 5V tolerant I/O pin must be used.



The 11XX series are sensitive to noise on the power and signal pins so clean power is important (with decoupling capacitors) and the signal line should be short and routed away from other signal lines.

The 16 Kbit version comes in two types with different UNI/O addresses and they can be connected in parallel using the same I/O pin. Depending on the command parameter supplied to the UNIO driver either can be accessed on demand. Any number of UNIO EEPROMs can be connected using other I/O pins to expand the capacity at low cost.

Usage

This CFunction will work on the Micromite (28 or 44 pins) and the Micromite Plus (64 or 100 pins). The CPU speed must be 20MHz or higher and the operation can take from 5 to 200 milliseconds (depending on the amount of data read/written) to complete.

The function takes five parameters and returns a status value.

```
status = UNIO(pin, cmd, addr, nbr, data)
```

The parameters are:

PIN	The pin number used to access the chip.
CMD	The command. This consists of two decimal numbers described below.
ADDR	The address of the first byte to be read/written (addresses in any one EEPROM start at zero and range up to 2048 (for the 16 Kbit version).
NBR	Number of bytes to read/write (described further below).
DATA	The variable holding data to be read/written. This can be a variable or an array. In the case of a write it can also be an expression.

The status value returned by the function will be 1 (or true) for success or 0 (false) if an error occurred. Errors can be caused by too low a CPU speed, no EEPROM detected, noise on the signal line, etc. The 11XX series of EEPROMs are sensitive to noise on the power supply or signal lines

so if the UNIO function returns zero you could retry the operation a number of times until you receive a true (indicating success) return. Also try reading/writing smaller amounts of data.

The CMD parameter consists of two decimal digits (for example 12). The first digit is the UNI/O address for the chip. Most chips use zero for this but the 11AA161 part number uses 1. Thus it is possible to parallel a 11AA160 and a 11AA161 on the same I/O pin and access them separately.

The second digit is the operation to perform as follows:

Second Digit	Operation
1	Read data from the chip.
2	Write data to the chip.
3	Erase the entire chip to all zero values The last three parameters of the CFunction will be ignored and can be left off (see the example below).
4	Erase the entire chip filling every byte with &HFF (all ones). As above, the last three parameters can be left off.

As examples:

01 will read from most versions of the 11XX chip (with UNI/O address of 0).

11 will read from a 11AA161 chip (with UNI/O address of 1).

02 will write to most versions of the 11XX chip (with UNI/O address of 0).

03 will erase the chip to all zero (assuming the chip has a UNI/O address of 0).

The NBR parameter will determine how many bytes will be written or read. For a floating point number this should be 4 and for an integer it should be 8. With integers it is possible to write less bytes if you are sure that the value of the integer will always be within a limited range. If the value of the integer will be 0 to 255 you only need to read/write one byte, for 0 to 65535 you can read/write two bytes and for numbers 0 to 4294967295 you can read/write 4 bytes. For negative numbers or if in doubt use the full 8 bytes for integers.

When reading/writing strings you can limit the number of bytes written to suit the maximum expected length of the string plus one. For example, if you know that the strings written will be 20 characters or less you can write 21 bytes (the extra byte is used by MMBasic to track the actual length of the string).

You can write arrays to the EEPROM by passing the array with empty brackets. For example dat(). You need to calculate how many bytes there are in the array and use this number for the NBR parameter. For example, if a floating point array is defined as DIM dat(10) you will need to write 44 bytes (arrays normally start from zero so there 11 elements with 4 bytes each).

Examples

Write the value of the floating point variable TmpRdg to address 1000 in an EEPROM on pin 15.

```
status = UNIO(15, 02, 1000, 4, TmpRdg)
```

The same only this time writing to a 11AA161 EEPROM (UNI/O address 1) connected to pin 15.

```
status = UNIO(15, 12, 1000, 4, TmpRdg)
```

Reading the value at address 4C (hex) into the floating point variable TmpRdg.

```
status = UNIO(15, 01, &H4C, 4, TmpRdg)
```

Writing the integer array dimensioned as IntArr%(5) to the same location.

```
status = UNIO(15, 02, &H4C, 48, IntArr%())
```

Writing the string Name$ which will have a maximum of 30 characters.

```
status = UNIO(15, 02, &H4C, 31, Name$)
```

Erase the entire chip to zeros. The last three parameters are not required and can be left off.

```
status = UNIO(15, 03)
```

UNIO CFunction

To add the UNIO command to MMBasic you need to insert the following code somewhere in your BASIC program (you can use copy and paste from this document). The exact spot is not important but at the end of the program is normal.

```
CFunction UNIO( integer, integer, integer, integer ) integer
    0000022D
    8CE30000 00C31821 ACE30000 40024800 0043102B 1440FFFD 00000000 8C880000
    00A84024 00061040 00621821 ACE30000 40024800 0043102B 1040000D 2402FFFF
    8C820000 00451024 1048FFF9 00000000 8C820000 00452824 10A80005 2402FFFF
    00661821 ACE30000 03E00008 2D020001 03E00008 00000000
    27BDFFC8 AFBF0034 AFBE0030 AFB7002C AFB60028 AFB50024 AFB40020 AFB3001C
    AFB20018 AFB10014 AFB00010 00809021 00A08821 00C0F021 00E0B021 8FB50048
    8FB00050 12A0003E 24020001 26B5FFFF 00069840 24170001 2414FFFF 16A00005
    92C70000 00073840 8FA2004C 10000003 00473825 00073840 34E70001 24040008
    00872807 30A50001 24A6FFFA 00063080 02463023 8E030000 40024800 0043102B
    1440FFFD 00000000 ACD10000 8E030000 02631821 AE030000 24A50005 00052880
    02452821 40024800 0043102B 1440FFFD 00000000 ACB10000 8E020000 02621021
    2484FFFF 1494FFE6 AE020000 40034800 0062182B 1460FFFD 00000000 AE510018
    02402021 02202821 03C03021 02003821 0411FF9D 00000000 16E00003 00000000
    18400007 00001021 12A00004 26D60001 26B5FFFF 1000FFC9 0000B821 24020001
    8FBF0034 8FBE0030 8FB7002C 8FB60028 8FB50024 8FB40020 8FB3001C 8FB20018
    8FB10014 8FB00010 03E00008 27BD0038
    27BDFFC8 AFBF0034 AFBE0030 AFB7002C AFB60028 AFB50024 AFB40020 AFB3001C
    AFB20018 AFB10014 AFB00010 00809021 00A09821 00C0A821 00E0F021 8FB40048
    00008821 00008021 2416FFFF 24170008 02402021 02602821 02A03021 0411FF6E
    02803821 1056002D 00118840 26100001 1617FFF7 02228825 3BC30002 0043F00A
    03C02821 27DEFFFA 001EF080 025EF023 8E830000 40024800 0043102B 1440FFFD
    00000000 AFD30000 24A50005 00052880 02452821 00151840 8E840000 00642021
    AE840000 40024800 0044102B 1440FFFD 00000000 ACB30000 8E820000 00621821
    AE830000 40024800 0043102B 1440FFFD 00000000 AE530018 02402021 02602821
    02A03021 0411FF44 02803821 0002102A 2403FFFF 10000002 0062880A 2411FFFF
    02201021 8FBF0034 8FBE0030 8FB7002C 8FB60028 8FB50024 8FB40020 8FB3001C
    8FB20018 8FB10014 8FB00010 03E00008 27BD0038
    27BDFFB0 AFBF004C AFB50048 AFB40044 AFB30040 AFB2003C AFB10038 AFB00034
    0080A021 00A09821 00C09021 8FB00064 8FB10068 24020055 A3A20020 8FA20060
    A3A20021 24020003 A3A20022 00071203 A3A20023 A3A70024 00061080 AFA20028
    24020005 AFA20010 24020001 AFA20014 27A20028 AFA20018 27A70020 0411FF2F
    00000000 10400015 00001821 12000013 24030001 2610FFFF 2415FFFF 27A20028
    AFA20010 02802021 02602821 02403021 0010382A 0411FF7D 00000000 10550007
    00001821 12000004 A2220000 26310001 1000FFF2 2610FFFF 24030001 00601021
    8FBF004C 8FB50048 8FB40044 8FB30040 8FB2003C 8FB10038 8FB00034 03E00008
    27BD0050
    27BDFFC0 AFBF003C AFB40038 AFB30034 AFB20030 AFB1002C AFB00028 0080A021
    00A09821 00C09021 8FB10050 24020055 A3A20020 A3A70021 24020005 A3A20022
    8E230000 40024800 0043102B 1440FFFD 00000000 AE930014 00121040 8E230000
    00621021 AE220000 24020003 AFA20010 24020001 AFA20014 AFB10018 02802021
    02602821 02403021 27A70020 0411FEEA 00000000 10400013 00001821 241001F4
    AFB10010 02802021 02602821 02403021 0411FF3D 24070002 04400007 30420001
    10400007 2610FFFF 5600FFF6 AFB10010 10000004 00001821 10000002 00001821
    24030001 00601021 8FBF003C 8FB40038 8FB30034 8FB20030 8FB1002C 8FB00028
    03E00008 27BD0040
    27BDFFA0 AFBF005C AFBE0058 AFB70054 AFB60050 AFB5004C AFB40048 AFB30044
    AFB20040 AFB1003C AFB00038 00809021 00A08821 00C0A021 00E09821 8FBE0070
    8FB50074 8FB60078 AFA00028 1AA00071 24020001 00061080 AFA20030 00061840
    AFA30034 24170055 00131023 00138027 2403FFF0 02038024 00508023 0010102A
    02A2800A A3B70020 A3BE0021 2403FF96 A3A30022 8FA20028 8FA30030 00431021
    AFA20028 24020003 AFA20010 AFA00014 27A20028 AFA20018 02402021 02202821
    02803021 0411FE9A 27A70020 10400051 00001021 A3B70020 2402FFA1 A3A20021
    2402006C A3A20022 00131203 A3A20023 A3B30024 8FA30028 40024800 0043102B
    1440FFFD 00000000 AE510014 8FA20034 00621821 AFA30028 24020005 AFA20010
    24020001 AFA20014 27A30028 AFA30018 02402021 02202821 02803021 0411FE7C
    27A70020 1040002C 02402021 AFB00010 AFA00014 27A20028 AFA20018 02202821
```

```
    02803021 0411FE72 02C03821 10400024 27A30028 AFA30010 02402021 02202821
    02803021 0411FF5C 03C03821 1040001E 02B0A823 1AA0001E 02709821 02D0B021
    8FA20028 40034800 0062182B 1460FFFD 00000000 AE510018 3C039D00 8C630000
    8C640000 24030B29 0083001B 006001F4 00002012 00441021 AFA20028 40044800
    0082202B 1480FFFD 00000000 AE510014 1000FF9E 00131023 10000006 00001021
    10000004 00001021 10000002 00001021 24020001 8FBF005C 8FBE0058 8FB70054
    8FB60050 8FB5004C 8FB40048 8FB30044 8FB20040 8FB1003C 8FB00038 03E00008
    27BD0060
    27BDFFB0 AFBF004C AFB50048 AFB40044 AFB30040 AFB2003C AFB10038 AFB00034
    00809821 00A0A021 00C08021 00E09021 8FB50060 24020055 A3A20020 32B100FF
    A3B10021 2402FF96 A3A20022 00061080 AFA20028 24020003 AFA20010 AFA00014
    27A20028 AFA20018 27A70020 0411FE1F 00000000 10400022 00001821 24020055
    A3A20020 A3B10021 A3B20022 8FA30028 40024800 0043102B 1440FFFD 00000000
    AE740014 00101040 00621821 AFA30028 24020003 AFA20010 AFA00014 27A20028
    AFA20018 02602021 02802821 02003021 0411FE06 27A70020 10400009 00001821
    27A20028 AFA20010 02602021 02802821 02003021 0411FEEF 02A03821 0002182B
    00601021 8FBF004C 8FB50048 8FB40044 8FB30040 8FB2003C 8FB10038 8FB00034
    03E00008 27BD0050
    27BDFFA8 AFBF0054 AFBE0050 AFB7004C AFB60048 AFB50044 AFB40040 AFB3003C
    AFB20038 AFB10034 AFB00030 00809021 00A08821 AFA60060 AFA70064 3C109D00
    8E020024 8C840000 0040F809 00002821 AFA20020 8E020028 0040F809 8E440000
    24030001 00431804 AFA30024 8E310000 2402000A 0222001A 004001F4 00008810
    0000F012 33DE0001 37DE00A0 8E020000 8C430000 3C020001 344286A0 0062001B
    004001F4 00002012 AFA40028 8E020014 8E440000 0040F809 00002821 8E020010
    8E440000 24050009 0040F809 00003021 8E02001C 8E440000 0040F809 2405000E
    8E020004 0040F809 2404000A 24130005 0000A021 24170004 24150002 24160003
    8E020014 8E440000 0040F809 24050001 8E020004 0040F809 240402BC 8E020014
    8E440000 0040F809 00002821 40944800 24020001 16220010 8FA20060 8C470000
    AFBE0010 8FA30064 8C620000 AFA20014 8FA40068 AFA40018 8FA40020 8FA50024
    0411FE49 8FA60028 1040001F 24040001 10000031 00002821 16350010 00000000
    8C470000 AFBE0010 8FA30064 8C620000 AFA20014 8FA40068 AFA40018 8FA40020
    8FA50024 0411FEBB 8FA60028 10400018 2673FFFF 1000001B 24040001 1636000A
    2407006D AFBE0010 8FA40020 8FA50024 0411FF41 8FA60028 1040000D 2673FFFF
    10000012 24040001 16370009 2673FFFF AFBE0010 8FA40020 8FA50024 8FA60028
    0411FF35 24070067 1440000A 24040001 5660FFBC 8E020014 00002021 10000006
    00002821 10000004 00002821 10000002 00002821 00002821 00801021 00A01821
    8FBF0054 8FBE0050 8FB7004C 8FB60048 8FB50044 8FB40040 8FB3003C 8FB20038
    8FB10034 8FB00030 03E00008 27BD0058
End CFunction
```

Adding the UNIO command to the Library

If you add this CFunction to the MMBasic library it will act exactly the same as a built in command and you will not have to add the above code to any program that uses this command. It will take up a minimum of memory and it will not be erased when you use NEW or load a new program. To do this you can follow these steps:

- Load the CFunction code listed above into the Micromite using either AUTOSAVE, XMODEM or the MMEDIT program. Do not include any other program lines unless you also want them stored in the library.
- Enter the command LIBRARY SAVE

This will transfer the CFunction code to the library area and delete it from the main memory. You can now use the command in your programs exactly the same as if it was a built in command. The only way it can be removed is by using the command LIBRARY DELETE or by reprogramming the chip with the Micromite firmware.