

WINDRUSH MICRO SYSTEMS LIMITED

UNIVERSAL EPROM PROGRAMMER, VERSION 2.05

THIS IS THE INITIAL RELEASE OF OS-9 LEVEL ONE SOFTWARE DRIVERS FOR THE WINDRUSH EPROM PROGRAMMER. IN ORDER TO REDUCE THE TIME REQUIRED TO PRODUCE A SET OF OS-9 DRIVERS WE HAVE TAKEN ADVANTAGE OF THE CAPABILITY OF OS-9 LEVEL ONE TO DIRECTLY ACCESS ANY MEMORY LOCATION WITHIN THE 64K MEMORY MAP. OS-9 LEVEL TWO DOES NOT ALLOW THIS TYPE OF MEMORY ACCESS. IN FACT OS-9 LEVEL TWO GOES TO GREAT LENGTHS TO PREVENT A PROGRAM FROM ACCESSING ANYTHING OUTSIDE OF ITS ASSIGNED MEMORY. THIS WAS DONE AS A PART OF THE LEVEL TWO MEMORY MANAGEMENT/PROTECTION SCHEME.

WHAT THIS MEANS IS THAT THERE IS NO EASY WAY OF RE-CONFIGURING THIS SOFTWARE PACKAGE TO RUN ON LEVEL TWO. WE ARE PRESENTLY WRITING A DEVICE DESCRIPTOR FOR OUR EPROM PROGRAMMER THAT WILL ENABLE IT TO BE USED ON OS-9 LEVEL 1 & 2 SYSTEMS. WE EXPECT TO HAVE A VERSION FOR LEVEL TWO UP AND RUNNING BY THE THIRD QUARTER OF 1982.

ANYONE WISHING TO PURCHASE THE LEVEL TWO VERSION SHOULD CONTACT US AFTER AUGUST 1, 1982. THE COST OF THE UPGRADE WILL BE \$25.00 WHICH INCLUDES REGISTERED AIR MAIL COSTS. U.K. AND EUROPEAN CUSTOMERS SHOULD CONTACT THE FACTORY FOR CURRENT PRICES.

THE FOLLOWING FILES SHOULD BE PRESENT ON THIS DISK:

1. UPROM\_OS9\_INFO.....THE FILE YOU ARE READING.
2. UPRM0.....OS9 EQUATES FILE
3. UPRM1D.....EPROM PROGRAMMER SOFTWARE PART ONE
4. UPRM2D.....EPROM PROGRAMMER SOFTWARE PART TWO
5. UPRM3D.....EPROM PROGRAMMER SOFTWARE PART THREE
6. UPROM.....AN OS-9 COMMAND (COPY TO THE 'CMDS' FILES)

TO ASSEMBLE THE FOUR SECTIONS OF THE SOFTWARE (2 THROUGH 5 ABOVE) DO THE FOLLOWING (ASSUMING THE ABOVE FILES ARE IN '/D1').

1. CHD /D1
2. DEL /D0/CMDS/UPROM .....ONLY NECESSARY IF UPROM ALREADY EXISTS ON /D0
3. ASM UPRM0 0=UPROM #10K

IF YOU RENAME THE PART 1, 2, OR 3 FILES YOU MUST CHANGE THE NAMES IN UPRM0 AS WELL.

THE MANUAL SUPPLIED WAS DEVELOPED FOR SINGLE USER (SSB DOS/FLEX) BASED SYSTEMS. THE BULK OF THE MANUAL APPLIES TO OS-9 SYSTEMS AS WELL. THE SECTION ON SOFTWARE CONFIGURATION/PATCHING CAN BE IGNORED AS THIS IS NOT REQUIRED.

ONE OF THE OPERATION COMMANDS HAS BEEN DELETED AND TWO OTHERS ADDED.

COMMAND '8', RETURN TO SYSTEM MONITOR, HAS BEEN DELETED FOR OBVIOUS REASONS.

A 'R' READ DISK FILE COMMAND AND A 'W' WRITE TO DISK FILE COMMAND HAVE BEEN ADDED TO ENABLE DIRECT ACCESS TO DISK FILES.

THE 'R' COMMAND WILL PROMPT YOU FOR:

1. NAME OF DISK FILE: .....Simply type in a standard OS-9 command line  
e.g. /dl/binfile\_in
2. A FILE OFFSET: .....Answer this prompt with a 4 digit Hex number  
which indicates how far into the file you  
wish to go before you start reading. For  
example if you wanted to start at the  
beginning of the file answer the prompt with  
\$0000. If you want to start 1K into the file  
answer the prompt with \$0400, etc.
3. A BUFFER OFFSET: .....Answer this prompt with a 4 digit Hex number  
which indicates at which point in the buffer  
you want to begin the load of the binary file.  
If you want to have the disk file to start  
loading at the beginning of the buffer answer  
this prompt \$0000. If you want to start 1K up  
from the beginning of the buffer answer the  
prompt with \$0400, etc.
4. TRANSFER SIZE: .....The number of bytes to be read from the disk  
file. \$0400=1K, \$0800=2K, \$1000=4K, etc.

THE 'W' COMMAND WILL PROMPT YOU FOR THE SAME INFORMATION AS THE 'R' COMMAND.  
THIS COMMAND WORKS VIRTUALLY IDENTICALLY TO THE 'R' COMMAND EXCEPT THAT  
THE BUFFER IS WRITTEN OUT TO A SPECIFIED DISK FILE. IF THE FILE DOES NOT  
EXIST THE FILE WILL BE CREATED. IF THE FILE ALREADY EXISTS IT WILL BE  
OPENED AND MODIFIED.

The read command ('R') allow you to read in small sections of a large binary  
file stored on disk. If, for example, you had a binary file that ran to  
some 32K of object code and you wanted to program 2K devices there are two  
equally valid techniques to use.

1. As you can only read 16K maximum into the buffer at any given time  
it would not be possible to read in the entire file. So for the  
first read (option 'R') specify a 'FILE OFFSET' of \$0000 and a 'BUFFER  
OFFSET' of \$0000 and a 'TRANSFER SIZE' of \$4000.  
This will read the first 16K of the file into the buffer area.

Now program the first 2K EPROM. When it is complete use the 'MOVE DATA'  
option ('1') to move the next 2K block down to the programming buffer  
which would lie between \$0000 and \$07FF for a 2K device. Answer  
the FROM prompt with \$0800, the TO prompt with \$0000 and the NUMBER  
of bytes prompt with \$0800. Program the second 2K EPROM. Use the 'MOVE  
DATA' option again to move the next 2K block down to the programming  
buffer (FROM \$1000, TO \$0000, NUMBER \$0800). Repeat this process until  
the first 16K of program is completed.

Use the READ option to read the second 16K of the disk file into the  
buffer area by answering the prompts: FILE OFFSET \$4000, BUFFER OFFSET  
\$0000, TRANSFER SIZE \$4000. Repeat the programming and move procedure  
outlined above until the second 16K of program is completed.

This approach minimises the number of disk accesses.

2. The second approach simply reads 2K segments of the disk file into  
the EPROM programming buffer area. Select option 'R', answer  
the FILE NAME prompt, FILE OFFSET prompt with \$0000, BUFFER OFFSET  
prompt with \$0000, and TRANSFER SIZE prompt with \$0800. Program the  
first 2K EPROM. When completed select option 'R' again. This time

answer the FILE OFFSET prompt with #0800...all other answers are the same. Repeat this process, increasing the FILE OFFSET by #0800 each time until all 32K is completed.

The write command allows you to make a disk binary file from an EPROM read into the buffer. It also allows you to build up a very lengthy disk file from several EPROMS. There is no technical limit to the size of the file you can make, however it is limited in practical terms by the capacity of the disk!

The process priority of the EPROM programmer software has been set very high. Therefore when the programming an EPROM the system will tend to exclude all other activities. As the length of time required to program most EPROMS is only 2-3 minutes this should not cause any problems with most users. If you do find it annoying try experimenting with the process priority statement in the equates file 'UPRMQ' and re-assembling the source files.

As this is an initial release we would very much appreciate any comments

good or bad. All bugs reported WILL be corrected as soon as possible, usually within two to three weeks of notification.

Please direct all comments to...

William C. Dickinson

Windrush Micro Systems Limited  
Worstead Laboratories  
North Walsham, Norfolk  
NR28 9SA

TEL: (0692) 405189