

ROM X^{ce}

Apple IIe, IIc

Video ROM Programming Guide

Revision 1.00 – October 2021

Dean Claxton, Jeff Mazur



Contents

Programming Overview	2
Required Tools/Equipment	3
Workflow	3
ROMXce Video ROM Hardware	4
24 Pin Video ROM.....	4
28 Pin Video ROM.....	5
Jumper Pad Configuration – Run Mode (Default)	6
Jumper Pad Configuration – Programming Mode.....	7
ROMXce Video ROM Contents	8
Bank 0 Fonts (Keyboard Switch OFF).....	10
Bank 1 Fonts (Keyboard Switch ON).....	13
Customising the ROM.....	16
Video ROM Overview	16
Data Structure	17
Recommended Software Tools	18
Quick Edits	19
Convert UnEnhanced ROM to Enhanced.....	19
Convert Enhanced ROM to UnEnhanced	19
Patching Fonts using Omnivore.....	19
Writing the ROM Image.....	20

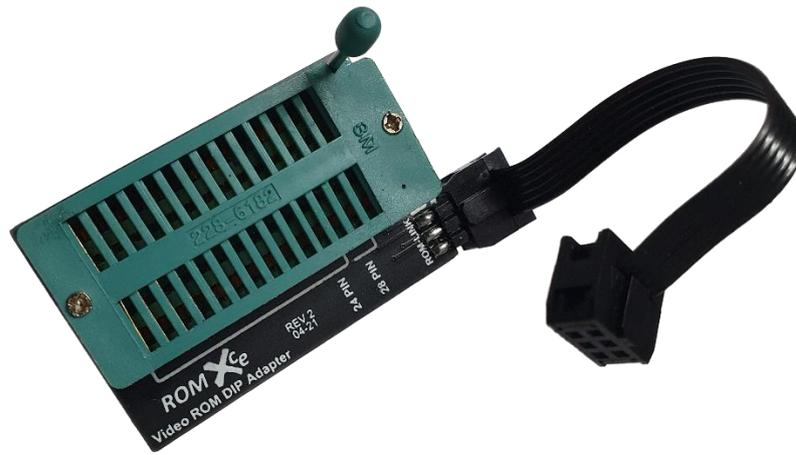
Programming Overview

This guide documents the methods and tools required to re-program the ROMXce Video ROM.

ROMXce Video ROMs ship with 32 fonts pre-installed (see the *Stock ROMXce Video ROM* section) however some of you may wish to further customise the font selection.

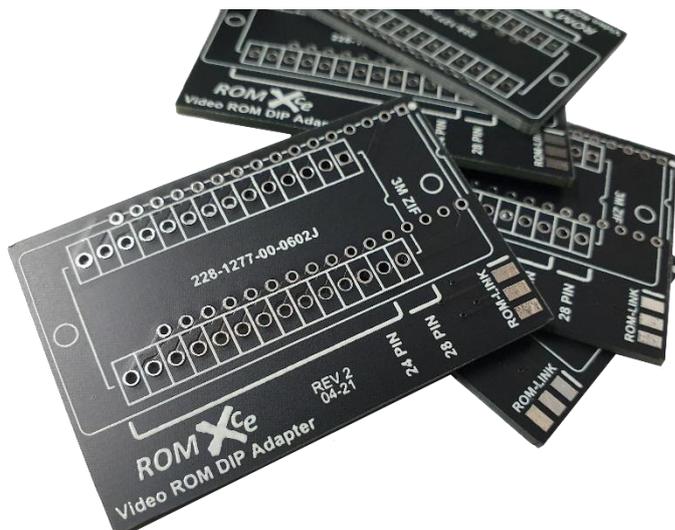
Whilst ROMXc and ROMXe are programmable in-system, the video ROM is not, and must be programmed using a ROMXce Video ROM DIP (dual in-line package) Adapter, and EPROM programmer.

The Video ROM DIP Adapter allows the surface mount 1Mbit SST39SF010A flash chip used on the ROMXce Video ROM to be programmed as a DIP package.



Manufacturing files (gerbers) for the programming adapter are available from <https://theromexcchange.com> in the documentation section along with this guide. These files can be submitted to a PCB fabricator should you wish to have your own PCB's manufactured.

Alternatively we do have blank PCB's available for a small fee, and can assemble the complete adapter on request.



Required Tools/Equipment

- Hex or graphical editing tools for manipulation of Video ROM data (see section *Recommended Software Tools*).
- Soldering Iron & solder (to re-configure the jumper pads on the rear of the PCB).
- Solder wick/braid or solder sucker (to remove solder from jumper pads on rear of the PCB).
- EPROM Programmer & software (must have support for SST39SF010A 1Mbit Flash chip).
- ROMXce Video ROM DIP Adapter.

Workflow

The workflow for customising a video ROM images is as follows :

1. Download the stock ROMXce Video ROM 1Mbit Flash Image from <https://theromexchange.com> – see the ROMXc/ROMXe documentation section.
2. Using the chart of flash memory offsets in the *ROMXce Video ROM Contents* section, locate the address of the font that you wish to modify or use as your baseline font.
3. Extract this image out to a separate 4KB file for editing.
4. Make your edits using your tool of choice (see section *Recommended Software Tools*)
5. Using the chart of flash memory offsets in the *ROMXce Video ROM Contents* section, locate the address of the font that you wish to replace with your custom font.
6. Slip your custom font into the master flash image, overwriting the previous. Ensure that you don't cross font address boundaries.
7. Prepare the ROMXce Video ROM for writing (see section *ROMXce Video ROM Hardware*)
8. Write your custom flash image (see section *Writing the ROM Image*)
9. Reconfigure the ROMXce Video ROM for normal operation and test.

ROMXce Video ROM Hardware

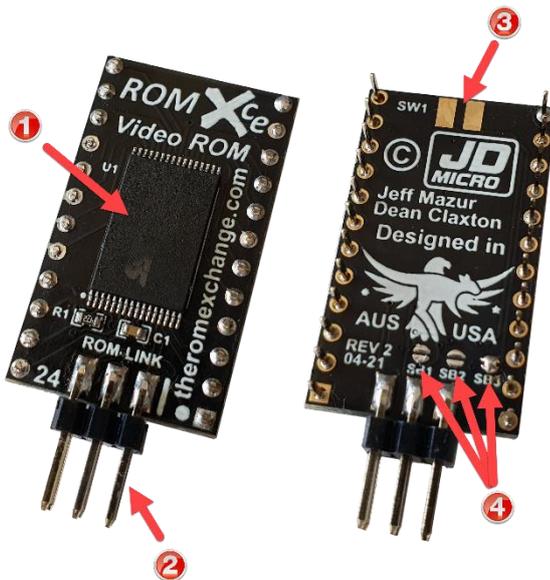
The ROMXce video ROM is loaded with 32 different fonts including both stock enhanced (MouseText) and unenhanced (no MouseText) fonts. The font list is covered in the user guide and is divided into 2 banks of 16 fonts. Bank selection is via external switch and the ROMX menu system allows selection of a font from the active bank to be used as the system default or associating a particular font with a ROM image. The video ROM connects to ROMXc/e via the supplied ROM-Link cable.

Two versions of the ROMXce Video ROM exist to accommodate the Apple IIc and Apple IIe – one for 24 pin and one for 28 pin video ROM sockets. Each ROMXce version has a slightly different mechanism for bank selection.

ROMXce Video ROM modules have jumper pads on the underside of the PCB that are used to select either programming mode, or normal run mode – see Jumper Pad Configuration.

24 Pin Video ROM

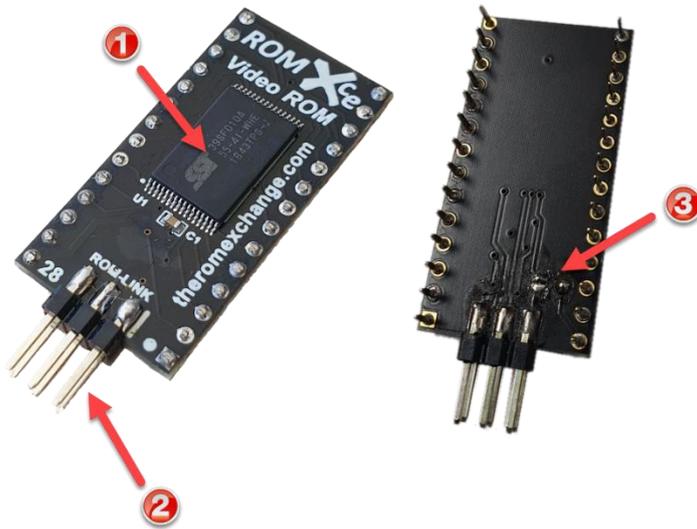
Bank selection is via a user installed switch or jumper on the SW1 pads.



1. 1Mbit Flash Memory.
2. ROM-Link Header – Pin 1 end (white bar on PCB).
3. Connection pads for external keyboard switch.
4. Jumper Pads SB1, SB2, SB3 used to select Programming or Run mode.

28 Pin Video ROM

Bank selection is via the stock keyboard switch.



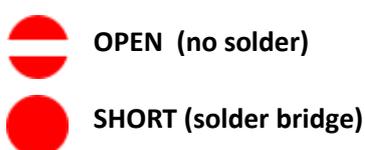
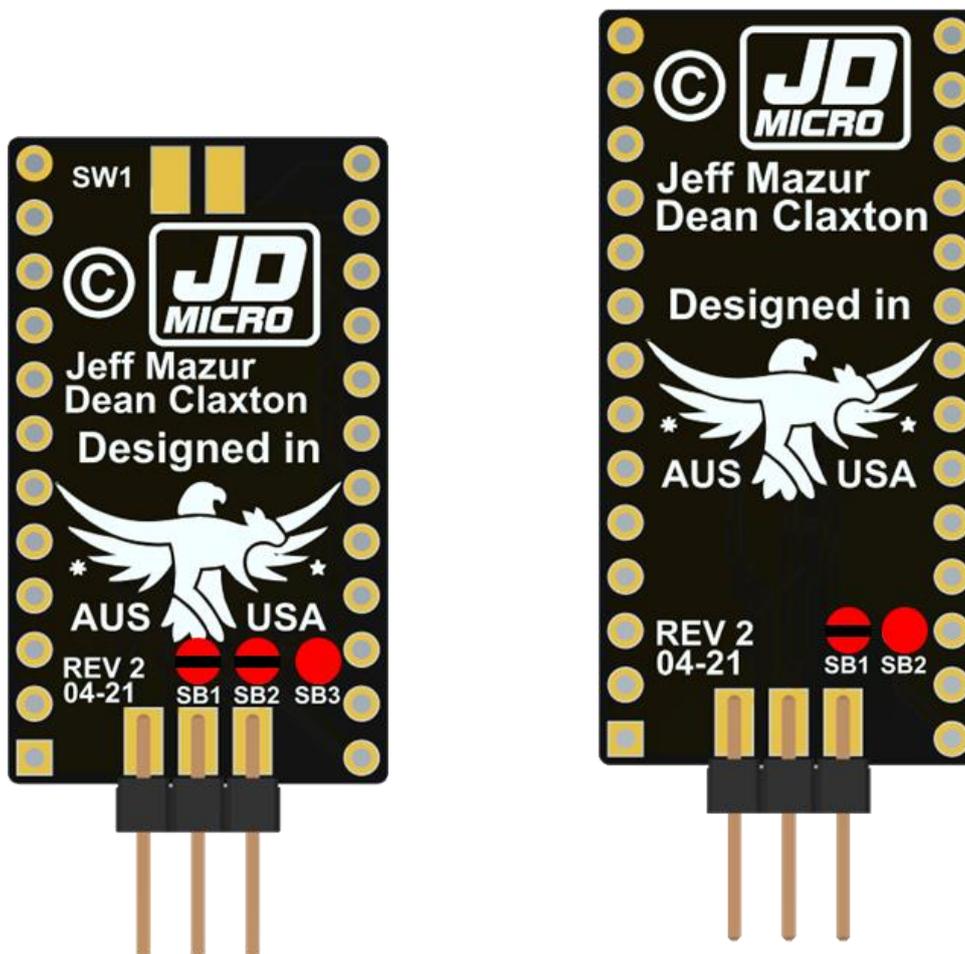
1. 1 Mbit Flash Memory.
2. ROM-Link Header – Pin 1 end (white bar on PCB).
3. Jumper Pads SB1, SB2 used to select Programming or Run mode.

Note that due to a manufacturing anomaly, the underside of the 28 pin video ROM is missing the overlay. Jumper pad SB1 is to the left when viewed as per above, and SB2 is to the right.

Jumper Pad Configuration – Run Mode (Default)

The Run Mode jumper pad configuration is shown below for the 24 pin module (left) and 28 pin module (right). This is the stock configuration.

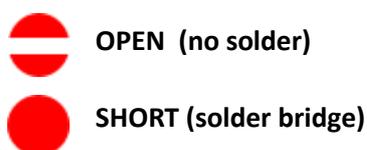
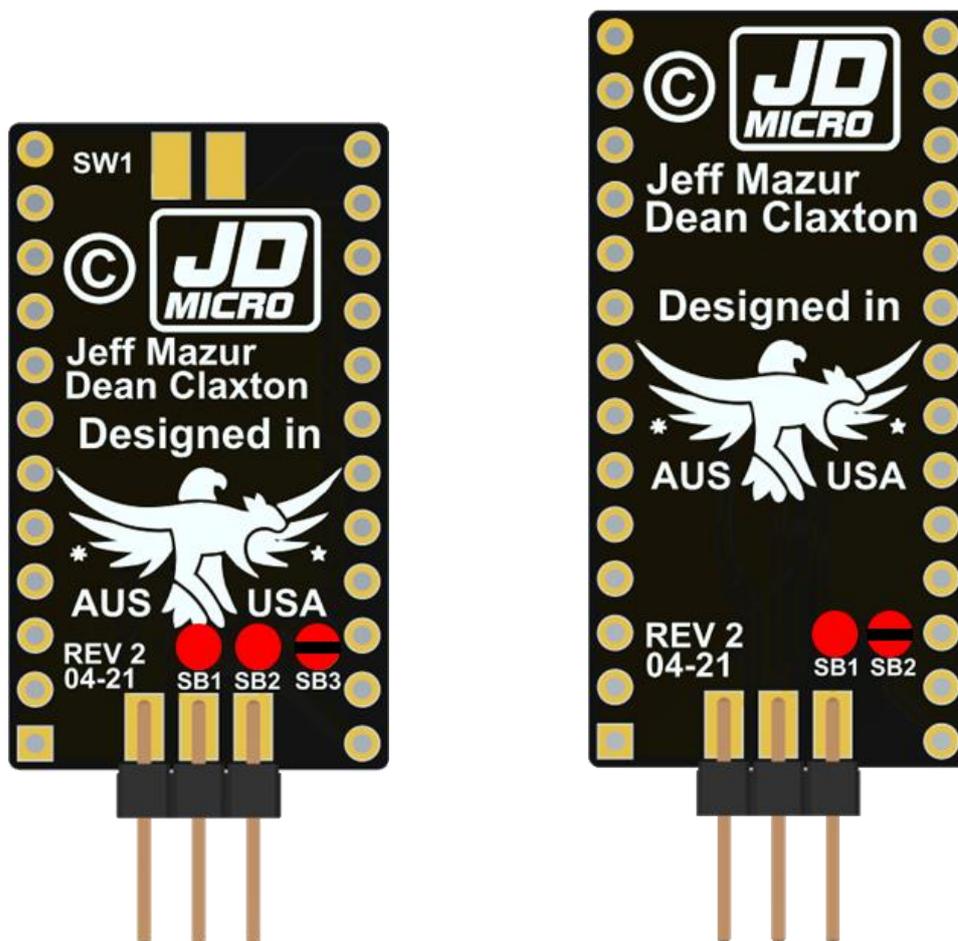
The pads shown as 2 half-moons are to be left open (unconnected) which can be done by user solder wick/braid to wick up the solder from the pad. The pad shown with the solid circle is to be shorted by placing a blob of solder over the half moon pads.



Jumper Pad Configuration – Programming Mode

The Programming Mode jumper pad configuration is shown below for the 24 pin module (left) and 28 pin module (right).

The pad shown as 2 half-moons is to be left open (unconnected) which can be done by user solder wick/braid to wick up the solder from the pad. The pads shown with the solid circle are to be shorted by placing a blob of solder over the half moon pads..



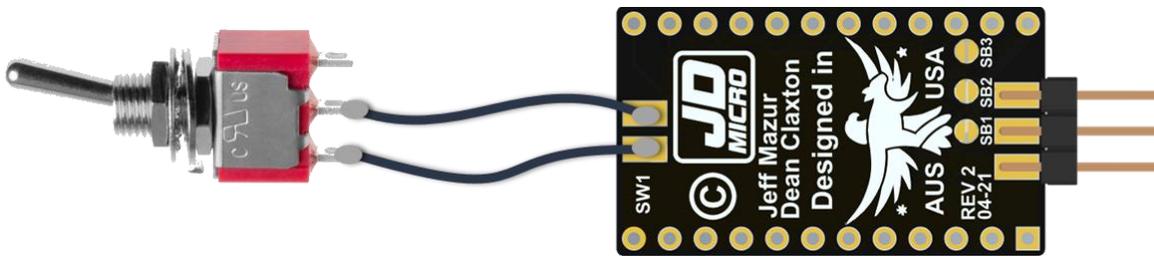
ROMXce Video ROM Contents

The ROMXce Video ROM contains a total of 32 different fonts, divided into 2 banks of 16. For Apple IIc and International Apple IIe (28 pin video ROMs), the stock keyboard switch will select the bank.

Note that the stock keyboard switch also selects a second keymap via the MAP ROM. On many international machines the two key maps stored in the MAP ROM are identical however for US based Apple IIc's the second key map is the Dvorak layout.

We have a ROMX MAP ROM Override board available that can be installed such that only key map 1 is used regardless of the switch position, leaving the keyboard switch to select only the video ROM bank.

For US Apple IIe with a 24 pin video ROM, an external switch can be installed:



The following table lists the fonts available. The font number is used in the ROMX menu system to associate the font. The memory offset in the SST39SF010A flash memory chip is also shown below.

ROMXce Font #	Keyboard Switch Off		Keyboard Switch On	
	Address	Font Name	Address	Font Name
0	00000	US Enhanced	10000	Euro Un-Enhanced
1	01000	US Un-Enhanced	11000	Euro Enhanced
2	02000	Clinton Turner V1 Enhanced	12000	Clinton Turner V2 Enhanced
3	03000	ReActiveMicro Enhanced	13000	Improved German Enhanced
4	04000	Dan Paymar Enhanced	14000	Improved German Un-Enhanced
5	05000	Blippo Black Enhanced	15000	French Canadian Enhanced
6	06000	Byte Enhanced	16000	French Canadian Un-Enhanced
7	07000	Colossal Enhanced	17000	Hebrew Enhanced
8	08000	Count Enhanced	18000	Hebrew Un-Enhanced
9	09000	Flow Enhanced	19000	Apple][Plus Enhanced
A	0A000	Gothic Enhanced	1A000	Apple][Plus Un-Enhanced
B	0B000	Outline Enhanced	1B000	Katakana Enhanced
C	0C000	Pigfont Enhanced	1C000	Cyrillic Enhanced
D	0D000	Pinocchio Enhanced	1D000	Greek Enhanced
E	0E000	Slant Enhanced	1E000	Esperanto Enhanced
F	0F000	Stop Enhanced	1F000	Videx Enhanced

Many of the included fonts are Enhanced, meaning that they include MouseText. MouseText is a set of 32 graphical characters designed by Bruce Tognazzini, which replaced a redundant set of inverse uppercase characters and made it possible to display simple text user interfaces resembling the Macintosh GUI.

The MouseText characters were introduced with the Apple IIc and later retrofitted to the Apple IIe via the Enhancement kit which included a 65C02 processor, the updated Video ROM and updated system ROMs. All Enhanced Apple IIe machines include MouseText, and MouseText was also carried over to the IIgs with a couple of changes (they replaced the running man characters shown in positions 7 and 8 below).

The 32 MouseText characters for the IIc and IIe :



And the IIgs version of MouseText, which is included in the ReActiveMicro video ROM:



We have included a fair selection of un-enhanced fonts for older applications that may have used the alternate inverse character set (and would look pretty terrible with MouseText).

Included on the ROMXce Utility Disk are a couple of Applesoft Text ROM Demo applications that allow you to use the arrow keys to preview each font, including the MouseText characters (see the Utility Disk section).

So let's run through the fonts that are included on the ROMXce Video ROM module.

Bank 0 Fonts (Keyboard Switch OFF)

0 = Apple US Enhanced Video ROM:



1 = Apple US Un-Enhanced Video ROM:



2 = Clinton Turner V1 Enhanced (Clinton based his font off the Commodore 64):



3 = ReActiveMicro Enhanced (Henry S. Courbis - includes the IIgs version of MouseText):



4 = Dan Paymar Enhanced:



5 = Blippo Black Enhanced (adapted from DOS toolkit):



6 = Byte Enhanced (adapted from DOS toolkit):



7 = Colossal Enhanced (adapted from DOS toolkit):



8 = Count Enhanced (adapted from DOS toolkit):



9 = Flow Enhanced (adapted from DOS toolkit):



A = Gothic Enhanced (adapted from DOS toolkit):



B = Outline Enhanced (adapted from DOS toolkit):



C = Pigfont Enhanced (Keith Comer):



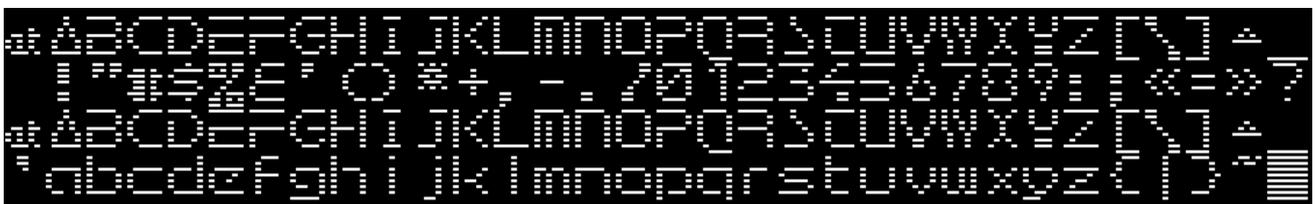
D = Pinocchio Enhanced (adapted from DOS toolkit):



E = Slant Enhanced (adapted from DOS toolkit):



F = Stop Enhanced (adapted from DOS toolkit):



Bank 1 Fonts (Keyboard Switch ON)

0 = Apple Euro Un-Enhanced:



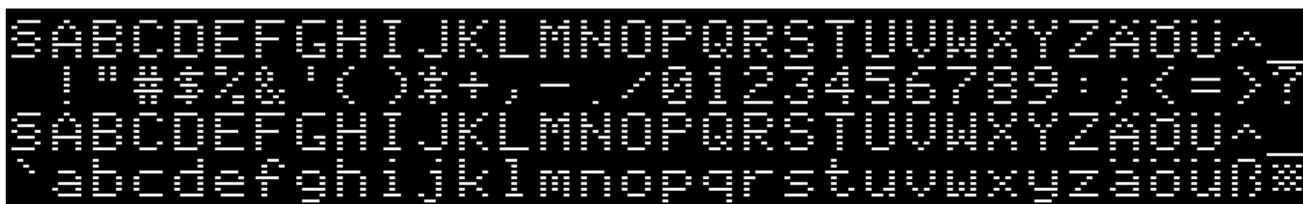
1 = Apple Euro Enhanced:



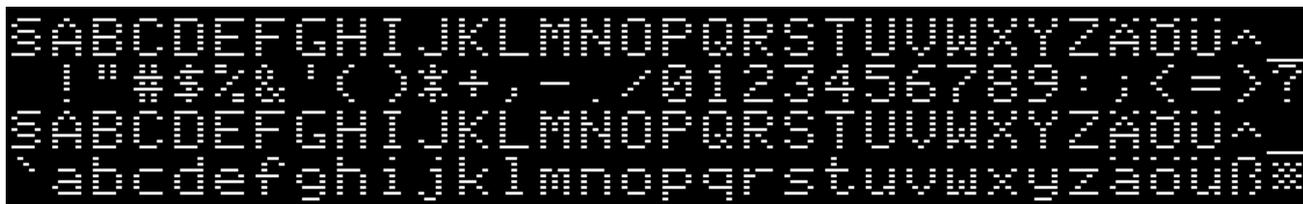
2 = Clinton Turner V2 (Clinton created this version for viewing on LCD displays):



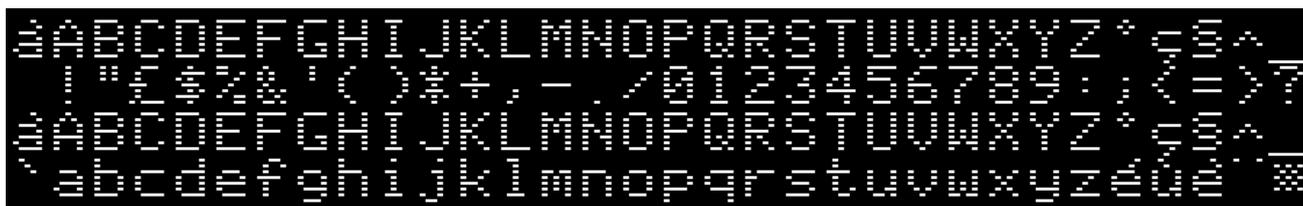
3 = Apple Improved German Enhanced:



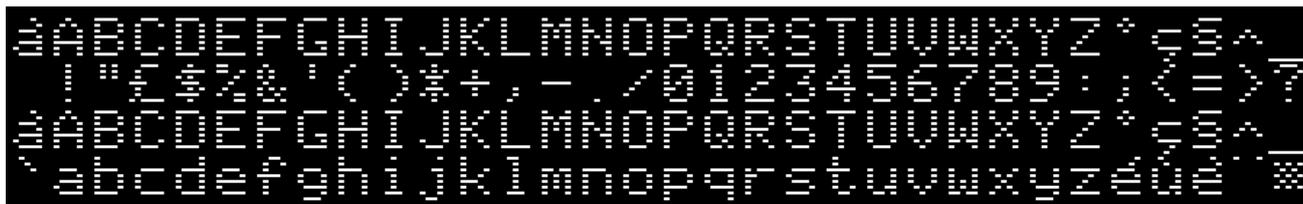
4 = Apple Improved German Un-Enhanced:



5 = Apple French Canadian Enhanced:



6 = Apple French Canadian Un-Enhanced:



7 = Hebrew Enhanced:



8 = Hebrew Un-Enhanced:



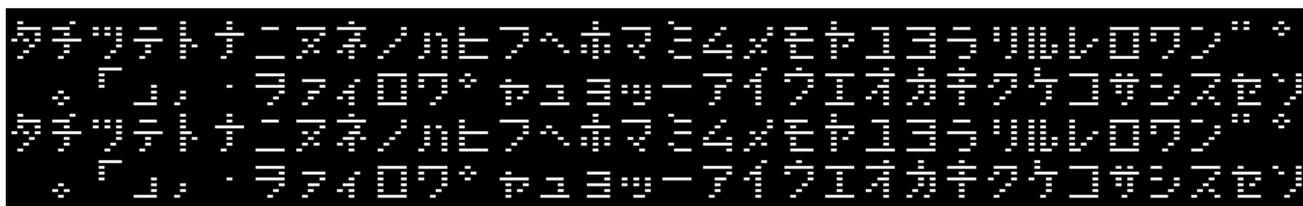
9 = Apple II Plus Enhanced:



A = Apple II Plus Un-Enhanced:



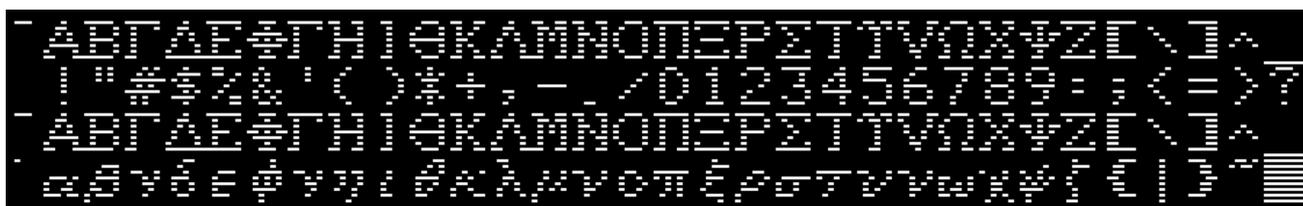
B = Katakana Enhanced (adapted from Apple J-Plus):



C = Cyrillic Enhanced (adapted from DOS toolkit):



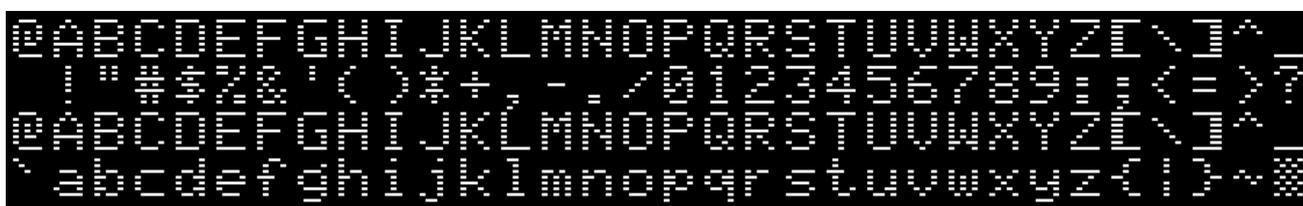
D = Greek Enhanced (adapted from DOS toolkit):



E = Esperanto Enhanced (adapted from DOS toolkit):



F = Videx Enhanced:



Customising the ROM

Video ROM Overview

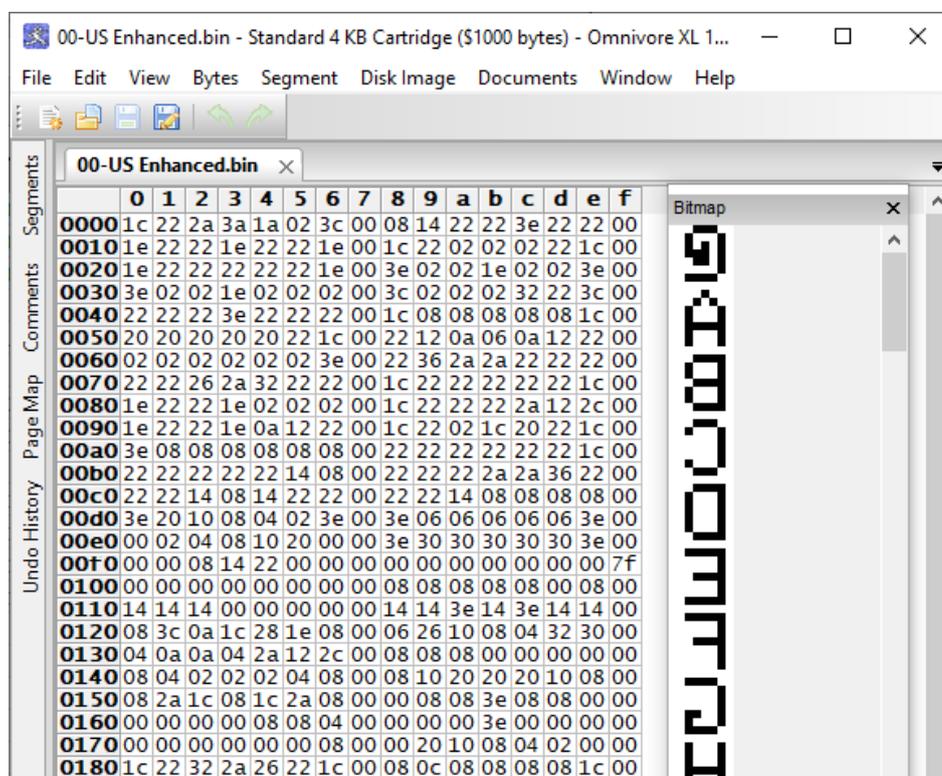
Before creating your own video ROM, we'd highly recommend the Video Generation section of Jim Sather's excellent book *Understanding the Apple IIe* as this will give you a good understanding of how the video ROM data is stored and displayed.

As an overview however, Apple IIc/IIe Video ROM images are 4KB in size (4096 (\$1000) bytes) with 8 bytes of data per character. Character data is stored in the lower 2KB (\$000-\$7FF), with graphics patterns stored in the upper 2KB (\$800-\$FFF). Coverage of the graphics patterns is outside of the scope of this document – generally you will simply copy that data from a stock Video ROM image.

Within the character data, each byte represents a row of character pixels however the data is stored with the rightmost character pixel represented by bit 6, and the leftmost character pixel represented by bit 0. Bit 7 is not used by system and for readability is typically left as a zero in the inverse area of the ROM and set to one in the normal area (see the Data Structure section).

The Apple IIe and IIc invert the character data when it is sent to the display – ones become black areas, zeros become lit pixels.

When viewing the data in an editor such as Omnivore (<http://playermissile.com/omnivore/>), the characters will appear to be backward, and will also be inverted versus what will actually be displayed on your Apple IIc/IIe computer.



Data Structure

The Apple IIc/IIe video ROM data is stored as follows :

\$000-\$1FF - ASCII \$00-\$3F (Inverse uppercase control characters including punctuation and numbers)

\$200-\$2FF - ASCII \$40-\$5F (MouseText on Enhanced ROM, else inverse uppercase for UnEnhanced ROM)*

\$300-\$3FF - ASCII \$60-\$7F (Inverse Lower Case)

\$400-\$5FF – ASCII \$80-\$BF\$ (Normal uppercase control characters)

\$600-\$6FF – ASCII \$A0-\$DF (Normal uppercase characters))

\$700-\$7FF – ASCII \$E0-\$FF (Normal lower case)

\$800-\$FFF – Graphics Patterns

** normally this would be a copy of \$000-\$0FF for UnEnhanced ROM*

Note that flashing text is created by the system switching between the \$000-\$1FF and \$400-\$5FF areas.

The Graphics Patterns stored in location \$800-\$FFF are out of scope for this user guide – you can copy these patterns from the stock Apple IIc/IIe video ROM.

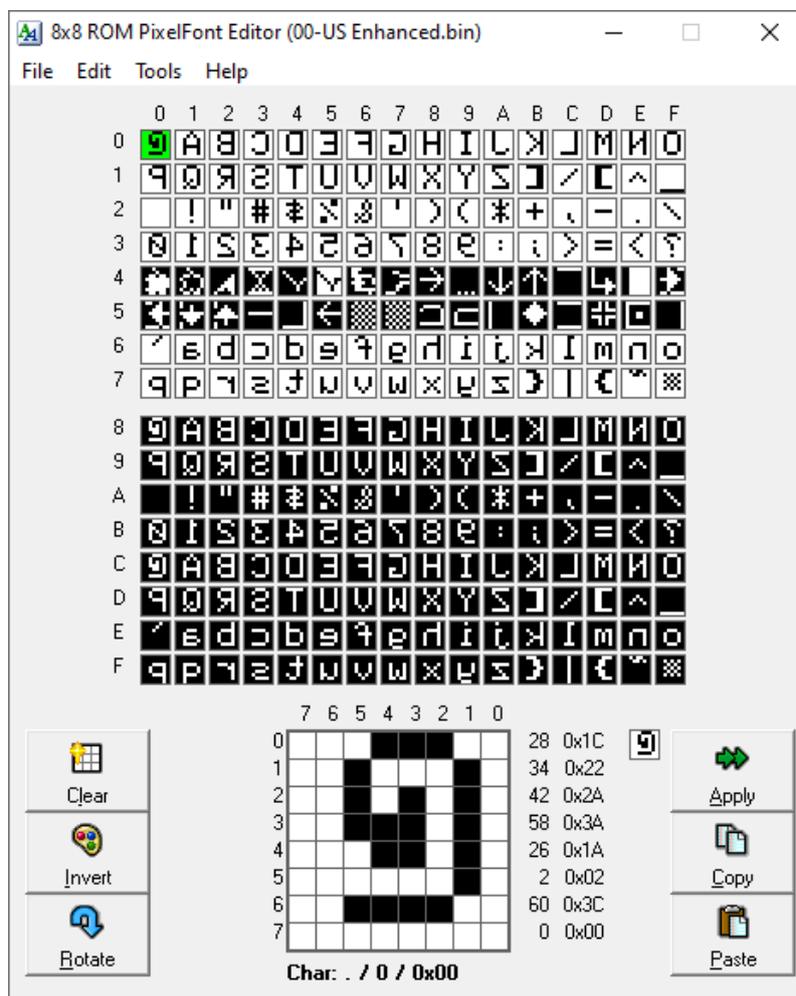
Recommended Software Tools

Omnivore (<http://playermissile.com/omnivore/>)

Excellent for bit level manipulation of the video ROM, with the bitmap preview. Great for moving characters around (such as inserting MouseText into UnEnhanced ROMs, and has a fair number of byte level functions such as invert, rotate, etc.

8 x 8 Pixel ROM Font Editor (<https://www.min.at/prinz/o/software/pixelfont/>)

Pixel editor – allows editing of pixels at the graphical level. Works great and leaves the graphics patterns (\$800-\$FFF) alone.

**010 Editor** (<https://www.sweetscape.com/010editor/>)

Great for slipping your custom 4KB ROM video back into the 1Mbit master flash image before flashing.

Quick Edits

Convert UnEnhanced ROM to Enhanced

Copy the MouseText data located at address \$200-\$2FF of the Enhanced ROM and paste into \$200-\$2FF of UnEnhanced ROM.

Convert Enhanced ROM to UnEnhanced

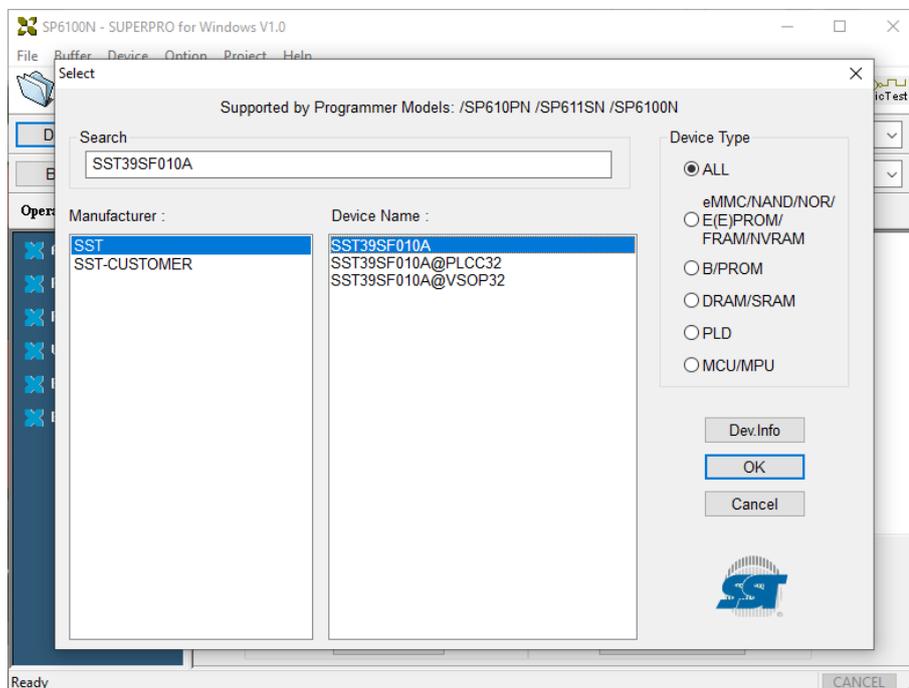
Copy the data from \$000-\$0FF of Enhanced ROM, and paste into \$200-\$2FF (overwriting the Mousetext)

Patching Fonts using Omnivore

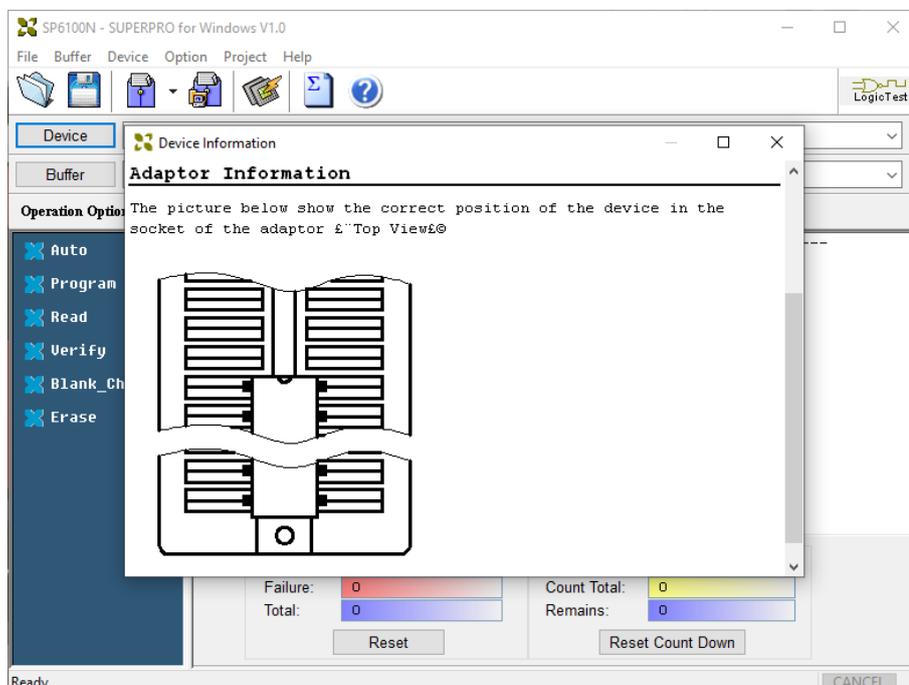
1. Modify uppercase inverse character set at \$000-\$1FF
2. Modify lowercase inverse character set at \$300-\$3FF
3. Copy \$000-\$1FF to \$400
4. Copy \$000-\$0FF to \$600
5. Copy \$300-\$3FF to \$700
6. Select bytes \$400-\$7FF, Bytes -> Bitwise NOT to invert.

Writing the ROM Image

Load your EPROM Programming software and select device SST39SF010A (DIP version) :



The EPROM Programming software will advise where the adapter must be positioned in the programming socket :



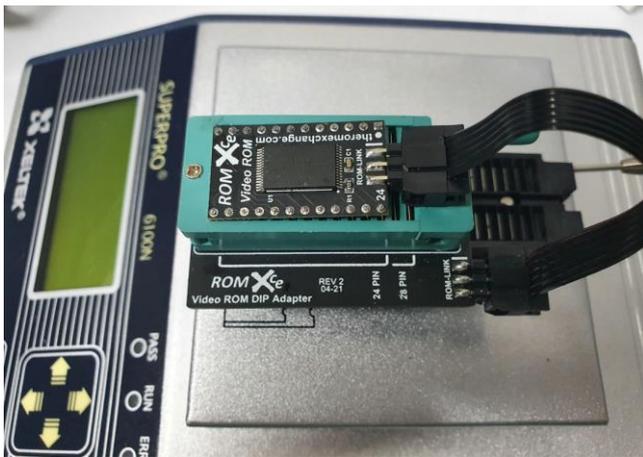
Note that the pin 1 position will vary depending on the model of your programmer!

Install the ROMXce Video ROM DIP Adapter into your EPROM programmer taking care to ensure it is positioned correctly as per the programming software guidance above. For Xeltek programmers, the adapter is positioned at the rear of the socket, but for other programmers it may be the opposite (TL866 for example is the opposite – the adapter is positioned in the front end of the socket).

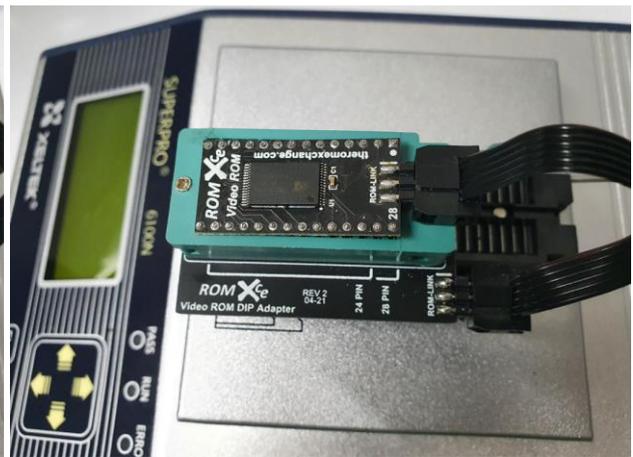


Connect the ROM-Link cable on the DIP adapter to the ROMXce Video and install it in the ZIF socket. The 24 pin Video ROM is positioned at the rear of the socket:

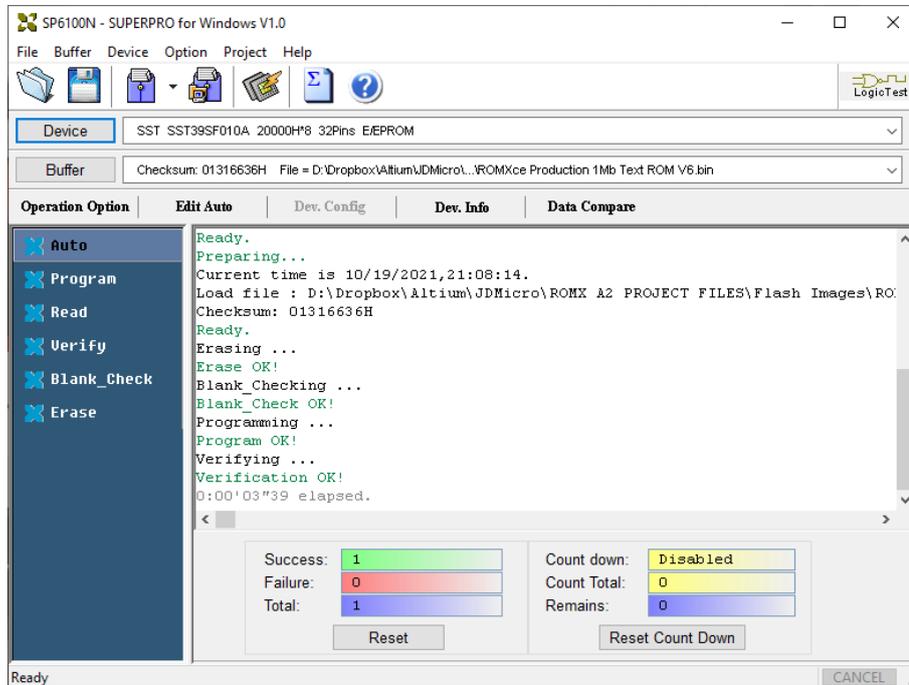
24 Pin Position



28 Pin Position



Once the Video ROM is in position, load your customised flash image into the programming buffer of your programming software and write it to the flash chip. Be sure to enable verification in your EPROM programming software.



If you receive any errors, double check your jumper pad configuration on the rear of the ROMXce Video ROM module and/or reposition the module in the ZIF socket and try again.

Once completed, reconfigure the solder jumper pads on the rear of the Video ROM for Run mode and test it in your computer.

