TECHTOOLS

*Flex*ROM+ User's Manual

(972) 272-9392

Fax: (972) 494-5814

email: sales@tech-tools.com

web: www.tech-tools.com

ftp: ftp.tech-tools.com

10-DAY Money Back Guarantee

IF, within 10 days of having received your product, you find that it does not suit your needs, you may return it for a refund. TechTools will refund the purchase price of the product, excluding shipping/handling costs,providing the product has not been altered or damaged.

Warranty

TechTools warrants its products against defects in materials and workmanship for a period of 90 days.

If you discover a defect,TechTools will, at its option, replace, repair, or refund the purchase price. Simply call us during business hours and ask for `Technical Support'. If we cannot solve your problem by phone, an RMA (Return Merchandise Authorization) number will be issued to you. Please include this number with the returned product.

This warranty does not apply if the product has been modified or damaged by accident, abuse, or misuse.

Copyrights and Trademarks

Copyright © 1994, 1995 by TechTools, All rights reserved. *Flex*ROM+, TechTools and all variations of the TechTools logo are trademarks of TechTools- Garland, Texas. Other brand and product names are trademarks or registered trademarks of their respective holders.

Disclaimer of Liability

TechTools is not responsible for special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, goodwill, damage to or replacement of equipment or property, and any costs or recovering, reprogramming, or reproducing any data stored in or used with TechTools products.

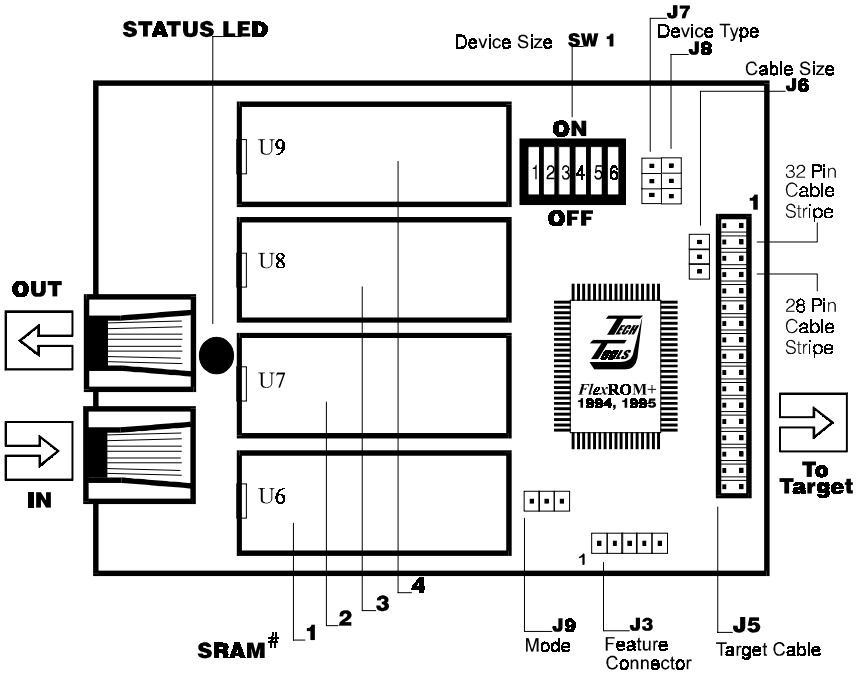
TABLE OF CONTENTS

INTRODUCTION

Thank you for selecting a TechTools product. We have made every attempt to provide a quality product at an affordable price. Our goal is to provide tools for Embedded Systems develoment that are inexpensive, but fully functional. If you have any problems or comments, please don't hesitate to call or FAX and let us know.

*Flex*ROM+ uses SRAM or battery-backed SRAM to emulate EPROMS and FLASH up to 4MBits in size. An IBM compatible computer (PC) is used to down-load object code into the SRAM.

CONFIGURATION



CABLE SIZE (J6)

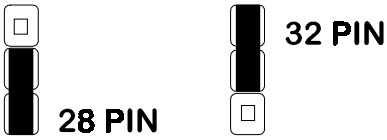


Figure 1-cable size (J6)

Set jumper J6 to indicate the size of the target socket. This jumper determines which target pin is used to supply power to the *Flex*ROM+. For 32 pin target cables, set the jumper to the "32" position.

For 28 pin target cables, set the jumper to the "28" position. The default configuration is for 32 pin targets (27010 or larger).

DEVICE TYPE (J7 and J8)

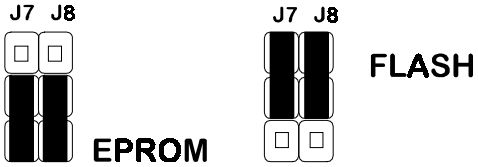


Figure 2-device type (J7, J8)

Jumpers J7 and J8 configure the FlexROM+ pinout to match EPROM or FLASH devices. Set jumpers J7 and J8 to indicate the type of target memory to emulate. If the FlexROM+ is emulating FLASH, set BOTH JUMPERS to the "FLASH" position. To emulate EPROM, set BOTH JUMPERS to the "EPROM" position. The default setting configures the FlexROM+ for an EPROM pinout.

OPERATING MODE (J9)

The MODE jumper (J9) determines the basic operating mode. This jumper configures FlexROM+ for Interrupt (IRQ) mode or Request/Grant (REQ) mode. In IRQ mode, FlexROM+ can generate interrupts to the target and generate wait-states for arbitration. In REQ mode, FlexROM+ can use bus REQUEST/GRANT handshaking for arbitration.

Figure 3-operating mode (J9)

If you want to use interrupts and/or wait-state arbitration, set J9 to "IRQ". If you want to use Request/Grant arbitration, set J9 to "REQ". If you are using none of these features, leave J9 set to "IRQ". This is the default configuration.

SRAM

Install the SRAM into the *Flex*ROM+. Insert a 1Mbit (128Kx8) SRAM (TC551001APL or equivalent) into location U6 with the notch pointing the same direction as the notch shown on the silk-screen. Additional SRAM, if used, should be inserted in order from U7 through U9.

EPROM/FLASH SIZE (SW1)

The dip switches configure the *Flex*ROM+ for the DEVICE SIZE being emulated. These MUST be set correctly for the unit to function properly. These switches intercept the upper address lines from the target. SW1-1 controls the highest address line (A18): The next one controls A17 and so forth. To emulate a 4Mbit device (27C040,28F040...), turn ON all switches. To emulate a 1Mbit device, turn OFF SW1-1 and SW1-2 ONLY. This continues until all switches are OFF for a 64Kbit device. This is summarized in Figure 4. (see also-CONFIGURATION)

SW 1 -Device Size Settings

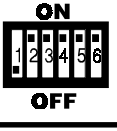
Device	1	2	3	4	5	6	Detail
2764	off	off	off	off	off	off	
27128	off	off	off	off	off	on	
27256	off	off	off	off	on	on	
27512	off	off	off	on	on	on	
27010	off	off	on	on	on	on	
27020	off	on	on	on	on	on	
27040	on	on	on	on	on	on	

Figure 4-device size (SW 1)

NOTE: “off ” is the “down” position

BASIC INSTALLATION

1. Turn power off to the Target.
2. IF YOUR TARGET HAS IN-CIRCUIT EPROM PROGRAMMING CAPABILITY, DISABLE IT!

   The *Flex*ROM+ is a 5 Volt ONLY device. Programming voltages will DAMAGE the device and will void the warranty. Even short surges during power-up or reset can be damaging.

3. Install target extension cable.

   Install the appropriate DIP cable on the *Flex*ROM+. If you are emulating a 28 pin EPROM, use the 28 pin cable. Otherwise use the 32 pin cable. In either case, the cable should be installed so that the colored stripe on the cable is closest to pin 1 on the emulator. Note that both cables terminate into a 34 pin connector. Mate this connector with the 34 pin header on the emulator.

4. Connect the *Flex*ROM+ to the target socket.

   BE CAREFUL TO INSERT THE *Flex*ROM+ CABLE PROPERLY. INSERTING THE CABLE BACKWARDS CAN RESULT IN PERMANENT DAMAGE TO THE *Flex*ROM+ OR THE TARGET SYSTEM!

   Pin 1 of the *Flex*ROM+ cable is identified by the colored stripe on the edge of the cable. Insert the cable so that this stripe aligns with pin 1 of the target connector. The target pin 1 is usually identified with a dot or a notch in the socket.

5. Select an un-used printer port on your IBM compatible PC/XT/AT.
6. Plug the supplied DB-25 to RJ-45 converter into the selected port.
7. Plug one end of the supplied RJ-45 cable into the adapter.
8. Plug the other end of the RJ-45 cable into the *Flex*ROM+ "IN" jack.

9. If you are daisy-chaining 2 or more units together, the “OUT” jack on the first unit should be connected to the “IN” jack of the second unit. In turn, the “OUT” jack of the second emulator would feed the “IN” jack of the third emulator, and so forth. Up to 4 emulators may be daisy-chained in this manor.

NOTE: The FIRST BYTE in the file will always end up in the LAST EMULATOR in the chain. This would be the emulator with only ONE cable attached to it. In the case of two emulators, the LAST on in the chain would hold the EVEN (0,2,4...) bytes and the FIRST one in the chain (closest to the PC) would hold the ODD (1,3,5....) bytes.

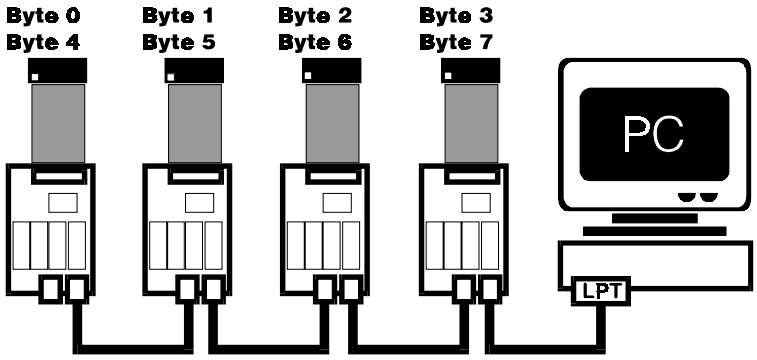


Figure 5-daisy chaining

10. (OPTIONAL) Connect a jumper wire between one of the reset pins (active high or active low) on the *Flex*ROM+ to the reset circuitry on the target if automatic reset is desired during each download. RESET and /RESET are tri-stated TTL/CMOS compatible outputs, capable of sourcing or sinking 15 milliamps. They are active during downloads and tri-stated during emulation. Refer to Table 1 below.

    *NOTE: The target MUST be reset after a download. This is done automatically if you connect the reset line from the emulator to the target. If you choose NOT to connect the reset line to the target, you MUST manually generate a reset after the download.

PIN	FUNCTION	NOTES
1	RESET	pin closest to LED
2	/RESET	
3	/WRITE	from TARGET
4	/WAIT or /BGNT	*See note below
5	IRQ or /BRQ	*See note below

Figure 6 - Feature Connector (J3)

*NOTE: Pins 4 and 5 are dual purpose. Thier function is determined by the setting of the “MODE” jumper.

11. Apply power to the target system.
12. Run the loader program. We wrote PlusLOAD.EXE as a command line driven routine rather than an interactive one. This allows you to run the program from a batch file without intervention. It can be added directly to your Compile-Link-Locate batch file.

    Enter “PlusLOAD” without parameters to see the parameter syntax and defaults.

    All parameters are optional and can be listed in any order. The Printer Port is specified by base address rather than LPTx. This allows the use of multi-port parallel cards that do not map directly into LPT1-LPT3. Most BIOS/DOS compatible printer ports are located at one of the following I/O addresses: 378, 3BC, or 278.

    You will also find “PlusEDIT.EXE”, a full-screen, interactive editor on the disk. PlusEDIT allows you to edit the file contents before downloading it. You can also make patches to the code after it is downloaded.

    For complete descriptions of these programs see-SOFTWARE-page 9.

ADDITIONAL NOTES

1. Most linker/locators will generate Binary files quicker than Intel Hex or Motorola 'S' files. Binary files are more compact (if your EPROM is over 40% full) and therefore download quicker. With that in mind, we wrote PlusLOAD to use Binary files. If your Linker or Locator will only generate HEX files, you will need to convert them to Binary before downloading them. The UTILITY directory on the DISK contains a utility that will do this conversion for you. HEX2BIN.EXE will accept Intel, Motorola and Tektronics HEX files in 8, 24 and 32 bit address formats and convert them to binary.

   Type HEX2BIN (without parameters) to see the parameter syntax.

2. The RESET and /RESET outputs are driven by TRI-STATED devices. They are active during the download only.
3. The Emulator (and therefore the target) must be powered before the object code can be down-loaded into it. If you use a battery-backed-SRAM, it will retain its information when power is removed. However, power MUST be applied to the *Flex*ROM+ before new data can be downloaded into it.

SOFTWARE

TECHTOOLS has provided two methods of operating your FlexROM+.

1."**PLUSEDIT.EXE**" - a full screen interactive interface for setup, pre-download editing and post-download editing/patching.

2."**PLUSLOAD.EXE**" - a command line "batchable" setup and downloader for time-saving automation.

PLUSEDIT.EXE

SETUP

Load "PLUSEDIT.EXE" from the DOS PROMPT. In the middle of the screen, the PLUSEDIT SETUP window will be displayed. The following options are available:

EPROM SIZE- This must accurately reflect the size of the device being emulated, and MUST NOT be set for a device larger than the ram installed on ***Flex*ROM+**. (NOTE: This should also match your DEVICE SIZE setting of SW-1, see Figure 4, page 4)

PORT- Selects parallel port address for the emulator. (if you need to use a different address than those listed, see "PLUSLOAD.EXE")

NUMBER OF ROMS- Selects the number of daisy-chained FlexROM+'s. Up to four units may be chained together. (see Figure 5, page 6)

SEMAPHORE- Controls the "semaphore" byte. The semaphore byte is a location in memory which can be accessed by the PC AND your TARGET, without arbitration. "LOW" configures FlexROM+ to decode offset 0 as the semaphore byte; "HIGH" (27040 only) instructs FlexROM+ to use 7FFFF as the semaphore byte. Any access to the selected location will access the semaphore byte instead of Normal memory. (see Advanced Installations- page 16)

Many customers do not need the Semaphore Byte, so the default setting is "DISABLE".

ARBITRATION- This determines how the emulator and your target respond to PC access (see ADVANCED INSTALLATIONS, pg. 16).

"NONE"- All reads, writes and downloads occur without regard to the current status of the target. This is not recommended because collisions will occur, possibly "locking up" the target.

"RESET"- Most common. FlexRom+ resets the target during reads, writes, and downloads. (see page 6)

"READY"- FlexROM+ and the target use the ready line to arbitrate.

"HOLD"- FlexROM+ and the target use the bus grant lines to arbitrate. (see OPERATING MODE (J9) , page 2; Figure 6 - Feature Connector (J3), page 7)

"AUTO UPDATE"- When selected, data changes made in the hex editor buffer, are automatically written to the emulator when you leave the current line. If not selected, updates will not take place until you request an emulator/write operation.

LOADING A FILE

To load a file from disk into the hex editor buffer, select OPEN from the FILE menu (or F3), and select your file. Next, choose your file TYPE. The next two options are as follows:

"START ADDRESS"- Loading address. With binary files, loading begins at the starting address and continues until all data is loaded into the buffer.

With an Intel or Motorola file, the START ADDRESS is subtracted from the address information present in the file. If a Hex file starts at 1000h, then the START ADDRESS should be set to 1000h.

"FILL VALUE"- Any value entered here will be used to fill unused locations in the emulated EPROM.

Once the file is loaded into the buffer, you will see the hex editor screen. This screen allows byte by byte hex or ASCII editing. For user control info, click on "Help" at the bottom of the screen (Alt-h). A window will appear explaining all editing keys.

READING from the Emulator

To read data from the emulator into the buffer, select READ from the EMULATOR menu. A window appears, asking you to wait while data is being read into the HEX EDITOR buffer.

DOWNLOADING to the Emulator

To download the current contents of the HEX EDITOR buffer into the emulator, select WRITE from the EMULATOR menu. A window appears, asking for the following information:

"BUFFER START"- This is the address in the HEX EDITOR buffer, where you want the download to start (usually 00000h).

"BUFFER END"- This is the address in the HEX EDITOR buffer, where you want the download to end. This is usually set to the highest address for the EPROM being emulated.

"OFFSET"- This is the address in the EMULATOR where you want the download to start.

As an example, if the first **255** bytes of the HEX EDITOR buffer need to be downloaded to address **1000h** in the emulator, the settings below would be used.

Buffer Start: 00000h

Buffer End: 000FFh

Offset: 01000h

Once the buffer and offset information have been entered, a window will appear, asking you to wait while the data is downloading to the emulator.

PLUSLOAD.EXE

If you want to automate your downloading, use Plusload.exe (allows direct downloading to the emulator, by-passing the hex editor buffer). This is a command-line driven, batchable downloader. Any number of options can be included on a single line, but they must be separated by spaces.

plusload file_name /option /option

All options and defaults can also be seen by typing "PLUSLOAD" from the DOS prompt, without parameters.

OPTION	DETAIL	DEFAULT
/P:	Printer port address in Hex 378	
/L:	Semaphore byte location	2
	0=LOW 1=HIGH 2=DISABLE	
/E:	Emulator Count 1-4	1
/v	Verify after download	no verify
/O:	Offset in hex	0
/M:	Arbitration Method	0
	0=RESET 1=READY 2=BUS GRANT	

FOR EXAMPLE, to use parallel port 278h and to request a verification after downloading, you would type **PLUSLOAD *filename* /p:278 /v.**

HEX2BIN.EXE

This batchable utility enables you to convert **Intel, Motorola** and **Tektronixs** Hex files to binary (supports 8, 24 and 32 bit formats). *For usage and options type HEX2BIN from the DOS command prompt.*

SHUFFLE.EXE

This batchable utility provides a way to interleave two binary files. From the DOS command prompt, type:

SHUFFLE (name of even file) (name of odd file) (name of output file)

example: **shuffle** even.bin odd.bin both.bin

DIVIDE.EXE

This batchable utility provides a way to break a binary file into two files. From the DOS command prompt, type:

DIVIDE (source file) (low file) (high file) (desired size of low file in hex)

example: **divide** big.bin low.bin high.bin 8000

TROUBLESHOOTING TIPS

PlusLOAD performs error checking before and (optionally) after the download. The possible error messages and possible solutions are listed below.

1. "A PRINTER PORT WAS NOT FOUND AT xxxx". At the beginning of the download, the program verifies that the device at the selected port address appears to be a printer port. This error is printed if there is no device at this address or the device does not respond like a printer port. Use the command line parameter "/P:xxx" to configure PlusLOAD for the base address of a valid printer port. The most common port addresses are 3BC, 278 and 378.
2. "ERROR - THE *Flex*ROM+ IS NOT RESPONDING .....". After the printer port is verified, PlusLOAD check to see if a *Flex*ROM+ is connected and communicating properly. This message is generated if the *Flex*ROM+ fails to respond properly. If you receive this message, check the following items:
    1. Verify that the *Flex*ROM+ is plugged into the printer port that was specified.
    2. Verify that the cable is connected properly
    3. Verify that the target is powered-up during the down-load
    4. Verify that the *Flex*ROM+ is plugged in properly
    5. Verify that J6 is jumpered properly for the target cable size.
    6. Remove any extention cables, switch boxs or security keys(dongles). Plug the supplied download cable DIRECTLY into the printer port.
    7 If you do buy a longer modular cable, be sure to purchase one that is wired the same as the one provided (8 conductor, pin 1 to pin 1). Note that modular cables can be purchased in straight through or cross-over configurations.
    8. Try a different printer port.
3. Once the port is verified and the *Flex*ROM+ responds properly, the download is completed. If the "/V" flag is used on the command line, the contents of the *Flex*ROM+ is then compared to the contents of the file that was just down-loaded. If a single byte is different, PlusLOAD will stop and report a "VERIFY ERROR AT xxxx". If you receive this error message, check the following:

1. Verify that the download file is NOT larger than the SRAM.
2. Have your SRAM checked, or try another one.
3. Remove any extention cables, switch boxs or security keys(dongles). Plug the supplied download cable DIRECTLY into the printer port.
4. If you extended the download cable, try removing it.
5. If you do buy a longer modular cable, be sure to purchase one that is wired the same as the one provided (8 conductor, pin 1 to pin 1). Note that modular cables can be purchased in straight through or cross-over configurations.
6. Try a different printer port or computer.

4. If the target does not respond as expected after the download, and you DID NOT receive any error messages, check the following:
   1. Verify that the DIP switches are set properly.
   2. Verify that J6,J7 and J8 are set properly.
   3. Verify that the download file will fit in the EPROM being emulated.
   4. Verify that the *Flex*ROM+ cable is plugged into the target properly.
   5. Verify that the target is supplying a full clean +5Volts to the *Flex*ROM+.
   6. Verify that the target has adequate power supply by-passing (particularly if it is a 2 layer or wire-wrapped board). The *Flex*ROM+ requires TWO to FIVE times as much current as the device it is emulating.
   7. Verify that the download file is a BINARY file. Use HEX2BIN to convert it if necessary.
   8. If you used a HEX to BINARY conversion program, verify that you specified the REAL physical starting address of the EPROM for the conversion.
   9. If you used the "/O" parameter for the download, verify that you really needed this option. It is rarely used.
   10. Use PlusEDIT.exe to view the contents of the emulator. Verify that the code is located properly and looks correct.

11. Try a manual reset of the target after the download. If this works, verify the reset polarity being used and that you are connecting to the correct place on the target.
12. Verify that the emulator access time (30ns + SRAM speed) is fast enough for the target.

ADDITIONAL HELP

If you need additional technical assistance, we can be reached at:

TechTools

PO Box 462101
Garland, TX 75046-2101
Voice (972) 272-9392 FAX (972) 494-5814 email:support@tech-tools.com

Please be prepared with the following information:

EPROM/FLASH device being emulated

DIP switch and jumper settings.

SRAM size and speed being used.

Command line parameters being used.

EXACT (if any) error messages generated.

Target information (CPU,SPEED,strange memory maps...etc.).

Anything that seems unclear or ambiguous in the instructions.

ADVANCED INSTALLATIONS

WE HIGHLY RECOMMEND THAT YOU VERIFY THE BASIC INSTALLATION BEFORE PROCEEDING TO THE MORE ADVANCED SETUPS.

This section documents the more advanced features of *Flex*ROM+ and how to use them. FlexROM+ supports three different Arbitration schemes and a DUAL-PORTED communications port. These mechanisms allows the target and HOST to each access the emulator's memory and to communicate without affecting each other; opening the door to several debugging techniques.

The arbitration mechanisms allow shared access to the emulator's memory. This capability can be used to implement a memory-mapped UART for HOST <-> Target communications. It also permits target programs to write debugging/trace information into un-used EPROM locations. The user can then read this information from PlusEDIT.EXE or with custom programs using PlusUTIL.LIB, without stopping the target.

In addition to arbitration, FlexROM+ has a single memory location that supports full dual-port accesses without any external arbitration support. We call this a communications port or a SEMAPHORE byte. This location can be set to offset 0 or offset 0x7ffff within the emulator's memory space. It can also be disabled (default).

The correct configuration for each of these features is documented below:

ABSOLUTE or RESET ARBITRATION

If no arbitration is selected, the *Flex*ROM+ will ALWAYS arbitrate for the HOST. This is the method used by the download program. If the target and HOST collide, the HOST will complete a valid cycle and the target will retrieve garbage. The RESET lines can be used to hold the target in reset during the operation, if desired. This is the fastest method of accessing the *Flex*ROM+ because it does not have to arbitrate for each access. If your application will need to be reset after the access (like in a code change) or if an occasional miss -read of the data is acceptable (as in some look-up tables) then this is the fastest and simplest method, requiring no extra connections.

CONNECTIONS:

MODE = IRQ
/WAIT - n/c
IRQ - n/c
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write (optional)

WAIT-STATE ARBITRATION

If the TARGET has wait-state circuitry, this can be used to effect arbitration. The /WAIT line on the *Flex*ROM+ is connected to the /WAIT line on the TARGET. The *Flex*ROM+ will wait for any TARGET access to complete before starting its access. If the TARGET then starts another access before the HOST access is complete, it will be held off with wait-states until the HOST access is complete. At that time, the /WAIT line is released and the TARGET is allowed to complete its access.

The /WAIT line is ACTIVE LOW, OPEN COLLECTOR.

CONNECTIONS:

MODE = IRQ
IRQ connected to TARGET IRQ (optional)
/WAIT - connected to TARGET /wait or ready
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write (optional)

Bus REQUEST/GRANT ARBITRATION

If the TARGET has bus mastering capabilities, this can be used for arbitration. The /REQUEST line from the *Flex*ROM+ can be connected to a bus request line on the TARGET. When the *Flex*ROM+ needs to access the *Flex*ROM+ memory space, it asserts /REQUEST. This tells the TARGET that a master wants control of the system. The TARGET arbitration circuitry holds off the local processor and generates a /GRANT signal. This signal is connected to the *Flex*ROM+ /GRANT pin. Since the processor is being held-off, it can not access the *Flex*ROM+ memory space and will not collide with the HOST access. When the HOST has completed its access, it will release the /REQUEST line, relinquishing control to the TARGET processor. (NOTE: IRQ is not available with this type of arbitration.)

CONNECTIONS:

MODE = REQ
/REQUEST connected to TARGET bus request
/GRANT connected to TARGET bus grant
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write (optional)

SEMAPHORE BYTE - AVOIDANCE

The *Flex*ROM+ contains a fully dual-ported byte of memory. This byte can be accessed by the TARGET and the HOST at any time without fear of a collision. A command-line flag allows you place this byte at offset 0 in the *Flex*ROM+ memory space or at offset 0x7ffff. The downloader DISABLES it by default. This SEMAPHORE byte can be used in several ways. The diagram below documents the contents of this byte.

SEMAPHORE BYTE DEFINITION

B7	B6	B5	B4	B3	B2	B1	B0
T1	T0	H2	H1	H0	C2	C1	C0

C0..C2 : COUNTER (READ ONLY)
H0..H2 : HOST CONTROL (Only the HOST can write, both can read)
T0,T1 : TARGET CONTROL (Only the TARGET can write, both can read)

In simple avoidance, the HOST accesses the *Flex*ROM+ at will. The target watches the counter bits to determine when it is safe to access the memory, thereby avoiding a collision. The counter bits are actually the bit counter for the serial download from the host. The HOST will actually READ or WRITE to the memory IMMEDIATELY AFTER THE COUNTER BITS ADVANCE TO xxxxx111B. If the target NEVER accesses the memory space when the counter bits are set, it will never collide with the HOST. This method does not require additional connections to the target. Naturally, this will NOT work if the target is executing out of this same memory space because the code fetches would not watch the counter bits.

CONNECTIONS:

MODE = IRQ
IRQ = n/c
/WAIT - n/c
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write (optional)

SEMAPHORE BYTE - HANDSHAKING

The control bits within the semaphore byte can be used to implement a handshake protocol between the HOST and TARGET.

For example, The HOST could set H0 to indicate that he wants control of the memory space and then wait for the target to grant access by setting T0. The TARGET will only do so when it is safe for him (he is executing out of some other memory space) for the duration of the HOST access.

The following pseudo-code illustrates this concept:

HOST:

```
char read_byte( address)
      set host request bit
       wait for target granted bit
       data = read(address)
       clear host request bit
       return(data)
```

TARGET:

```
void POLL(void)
      if (HOST request bit set)
         call GRANT
      return

// this routine MUST be outside of the emulator memory space
 GRANT(void)
      if (I feel generous)
                set my granted bit
                wait for HOST request bit to clear
                clear my granted bit
      return
```

CONNECTIONS:

MODE = IRQ
IRQ - n/c
/WAIT - n/c
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write

SEMAPHORE - HANDSHAKE with interrupts

This approach is similar to the last one. The main difference is that the *Flex*ROM+ /IRQ line is used to interrupt the TARGET when access is needed. A target SEMAPHORE bit is then used to indicate when access is granted. This eliminates the need for the TARGET to constantly poll for requests. If the interrupt routine is located in a memory space out-side of the emulator space, it could immediately grant the HOST access to the memory.

CONNECTIONS:

MODE = IRQ
/IRQ connected to TARGET interrupt
/WAIT - n/c
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write

SEMAPHORE - SOFTWARE UART

Use the semaphore bits to implement a software UART. In this approach, the HOST never accesses the memory space directly (except during the initial download). Commands are sent serially through the semaphore byte to the TARGET. The target responds serially back through the semaphore byte. The routines could use 1 bit for data and another bit for handshake (data ready/data received). This method is particularly suited for micro controllers that can not execute out of other memory space, do not have ready/wait lines and do not have bus request/grant lines. The only requirement is that they can WRITE to the EPROM space.

Since the memory space is not being accessed directly by the HOST, there is no need for arbitration. The IRQ line could be used to interrupt the TARGET when a valid data bit is ready from the HOST.

CONNECTIONS:

MODE = IRQ
/IRQ connected to target IRQ (optional)
/WAIT - n/c
/RESET or RESET connected to TARGET reset (optional)
/WRITE connected to TARGET /write

SUMMARY

The SEMAPHORE UART implementation provides hardware independent communications between the HOST and TARGET. It requires NO ARBITRATION connections and will work with any TARGET that can write to this memory space. However, it requires software on both ends of the link.

The remaining SEMAPHORE implementations rely on minimal software, but require that the TARGET can execute from code space outside of the memory space occupied by the *Flex*ROM+ during the HOST accesses. Many micro-controllers have a limited amount of internal memory that could be used for this. Other systems could place a small amount of code in RAM and execute from there.

The ARBITRATION (SHARED MEMORY) implementations are completely transparent to the software on both ends and allow each completely random access to the entire memory space at will. However, they depend on hardware support from the TARGET and one or two additional connections.

ADDITIONAL NOTES

The active levels and drive types (TTL/OC) of /REQUEST, /GRANT and /WAIT were selected for common applications. If your target requires different drive levels or drive types, you may need to add external logic.

LIBRARY REFERENCE

Refer to the "PlusUTIL.DOC" file on the distribution diskette for full documentation of the PlusUTIL libraries.

APPENDIX A

EMULATING 16 bit EPROMS

Two FlexROM+s can be combined to emulate 16 bit EPROMS with an ADP16 adapter. One emulator holds the ODD data (D8-D15) and the other one holds the EVEN data (D0-D7). When used in this configuration, the FlexROM+s and PlusEDIT need to be configured as follows:

FlexROM+ configuration:

Configure EACH FlexROM+ for ½ of the total device capacity. For example, if you are emulating a 27C220 or 27C2048 (2Mbit), configure EACH FlexROM+ for 1Mbit (SW1 and 2 off, 3-6 off).

PlusEDIT configuration:

Configure PlusEDIT for 2 EMULATORS, each of ½ the total device capacity. For a 27C220 or 27C2048, you would configure PlusEDIT for 2 - 27010s.

Connect the FlexROM+s to the ADP16:

Daisy-chain the two emulators together per the manual instructions. Connect the LAST emulator in the chain to the D0-D7 connector on the ADP16. Connect the remaining emulator to the connector marked D8-D15 on the ADP16.

Table of Figures

TECHTOOLS

(972) 272-9392

Fax: (972) 494-5814

email: sales@tech-tools.com

web: www.tech-tools.com

ftp: ftp.tech-tools.com