****
C L E A R L I G H T I N C .
****

----- A M P L / M -----

ADVANCED MULTI - IMAGE

PROGRAMMING LANGUAGE

MULTI-TASK VERSION

COPYRIGHT (C) 1981

R E F E R E N C E M A N U A L

A M P L / M

REFERENCE MANUAL

TABLE OF CONTENTS

CLEAR LIGHT INC.

-------- A M P L / M --------

ADVANCED MULTI - IMAGE

PROGRAMMING LANGUAGE

MULTI-TASK VERSION

COPYRIGHT (C) 1981

R E F E R E N C E M A N U A L

SECTION I

SYSTEM OVERVIEW

A. Hardware Configuration

The AMPL/M Multi-Image programming system operates on any Apple II 48k computer system with at least one disk drive, and DOS 3.3 installed.

Clear Light provides a slightly modified version of the Apple, called the Clear Light SUPERSTAR. The main differences between the SUPERSTAR and a standard Apple are: different color, different label, special key caps, the Clear Light warranty, a shift-key modification, and a fully installed and tested Clear Light Interface card.

A special version of AMPL/M is available for the 16k RAM card. The standard configuration allows approximately 2000 cues, while the 16K RAM card configuration allows about 6000 cues.

While it is not required, a second disk drive is extremely helpful, especially when making disk copies.

B. The Clear Light Interface Card

The AMPL/M software package includes a special interface card, which interfaces the computer to the tape recorder and to either a Star-3 Programmer or a Star Universal Interface. In addition, the card supplies a proprietary hardware lock, which is keyed to the specific version of AMPL/M provided with the card. The AMPL/M system will not operate without the interface card with a matching serial number.

This lock system provides the user with a very substantial benefit: he may make as many backup copies of his master disk as he needs, with no fear of copyright infringement. Obviously, these copies are to be used by only those persons who have direct access to the interface card as well. It is illegal to provide copies of the AMPL/M software to others for any purpose other than to program

multi-image programs, using the matching interface card. In fact, it is actually illegal even to attempt to break the software lock on the AMPL/M system, even if the attempt is a failure! Please reread the Software License Agreement that you signed when you purchased the AMPL/M system. It is important to be aware of the limitations of use specified in that document.

Enough of the heavy stuff! A problem you may be worried about is how to use your programs on someone else's SUPERSTAR. The need for this will occur often in the rental business. There is really no problem at all here: while the AMPL/M System Master disk and all its copies will only work with a specific interface card, once the system is running, it can read files from any other AMPL/M disk. Thus, when you rent or borrow someone else's SUPERSTAR as a backup or while traveling, make sure you get a copy of their System Master disk to run on their system.

C. Playback Targets

The AMPL/M software will program for playback through three different hardware configurations, called TARGETS. These targets are described below:

1. SUPERSTAR target: This mode uses the SUPERSTAR computer to run the show directly, usually from clock track, or, if live, with manual cues (or both). In this mode, the SUPERSTAR can operate up to 100 cues/sec, and may use a Star-3 Programmer, a Star Universal Interface, or a Star Controller for 15 projector operation. For 30 projector operation, two Star Controllers or two Star Universal Interfaces must be used.

2. CUE SENTRY target: This mode requires one or two Star Universal Interfaces to record the CUE SENTRY data track, and to play back the show. Up to 50 cues/sec is allowed on each Interface. Any of the hardware options specified in (1) above may be used to program the show. This means that relatively inexpensive Star Controllers may be used to program the show, without tying up the much more expensive Star Universal Interfaces.

3. STAR DISSOLVE target: This mode requires either a Star-3 Programmer or a Star Universal Interface to record the cue track, once the show is programmed. Any of the hardware options specified in (1) above may be used to program the show. The show then plays back on up to 5 Star-3 Dissolves, at a maximum speed of 10 cues/sec.

D. System Structure

The AMPL/M system is designed to be easy to use, by the use of HELP MENUS and a logical SYSTEM STRUCTURE.

The system structure is based on a series of levels, each of which have a menu associated with them. The user normally can see what his options are at any given level by calling up the associated

menu.

The top level of the system is called the MAIN MENU. From this menu, the user can move down one system level by selecting one of the menu options.

The user may "escape" from any level to a higher level by pressing the <ESC> key. The computer will respond with a very short, high pitched beep, indicating the shift to a higher system level. This beep is easily distinguished from the normal illegal entry beep by both its tone and duration. Usually, the <ESC> option is displayed on the bottom of the screen, or on the associated menu, to remind the user how to exit from that level.

While all this may seem confusing at first, you will find it becomes second nature very quickly, once you have had an opportunity to actually try the system.

E. Nomenclature

There are several conventions used in this manual to express user input into the computer. All keyboard entries are contained in angle brackets. For example, if you are to type the letters 1 O P F into the computer, this is written as <1OPF> in this manual.

As mentioned earlier, Clear Light sells a special version of the Apple][Plus, called the SUPERSTAR. The SUPERSTAR keyboard has two keys specially marked: CUE, and HELP. These special keys are the same as the RETURN and ? keys, respectively. In this manual, we will often refer to these keys as the HELP key and the CUE key.

Another area of confusion is the CTRL key. CTRL stands for CONTROL, and the key is used as a shift key. This means that the key does not cause any input itself (just like the SHIFT key), but rather modifies the key you type into a control key. Thus, if you press and hold down the CTRL key, and press the X key, this enters a CONTROL-X character into the computer. This is NOT the same as a regular X! This entry is written as <CTRL-X> in this manual.

At many points in this manual, we will refer to other sections of the manual for more information on a specific point. After a while, you will probably get tired of reading them! However, this information is included in the manual to help you find out more about specific functions or features, when you are using the manual as a reference. Also, it will assure the first-time reader that more information is forthcomming on particular points of interest. So don't be put off by the constant reference to other sections of the manual! Just ignore them, until you need to use them to find more information on a specific item.

SECTION II

GETTING STARTED

A. Connecting Up The Equipment

Included with your SUPERSTAR computer are a set of Apple manuals, which you will need to look at for installation instructions. Read Chapter 1 of the Applesoft Tutorial, and Chapters 1 and 2 of the DOS Manual. This will teach you most of what you need to know for now about disks and setting up your computer.

Now you are ready to install the Clear Light Interface Card, if it is not already in your computer. The card MUST be installed in slot #4, so that the jacks are available to the rear panel, and the software can operate properly. If you are installing the card yourself, refer to the instructions provided with the card.

One other adjustment you should make: Your keyboard has a small slide switch on its printed circuit board, which must be in the correct position to prevent RESET from operating unless you are also pressing the CTRL key. Since RESET completely stops the computer in its tracks, and forces it to enter a special interrupt program (described at the end of this section), you don't want to press it accidentally! By combining the RESET function with the CTRL key, this is extremely unlikely. Directions for this switch setting are included with your computer.

B. Booting the AMPL/M System

Turn off your computer, and place the AMPL/M System Master Diskette in disk drive #1. Turn on the power. The disk "in use" light should come on, the disk drive will "buzz", and then begin to load the system. First, the words "CLEAR LIGHT SUPERSTAR" will appear directly beneath the "APPLE][" title. Then the high resolution picture of the Clear Light logo will appear. A few seconds later, the DATE/TIME menu will appear. Simply enter the date and time as directed. You may use single or double digits for each entry, separated by a <SPACE>. You may also use the backspace (<-) key to backup and change any incorrect entry. You will even be given a last chance to change your entries, or to continue with the system boot. It is very tempting at this point to type in garbage for the date and time. Restrain yourself, and type in the correct date and time! The computer will use this information to help you keep track of your disk files, by saving the date each file was last stored to disk. This wonderful feature will be useless if the dates and times you enter are not correct. More on this special feature of the AMPL/M Disk Operating System is given in section XII of this manual.

Once you have entered the correct date and time, simply press <RETURN> to continue the boot. A few seconds later, you will be presented with the AMPL/M MAIN MENU, shown below:

```
!     C L E A R   L I G H T   I N C      !
!                                        !
!        ------------------------        !
!        S U P E R S T A R               !
!        ------------------------        !
!                                        !
! ------------ A M P L / M ------------ !
!                                        !
!                (C)  1981               !
!                                        !
!                                        !
!SELECT FUNCTION FROM MENU:#             !
!                                        !
!     N - NEW (ENTER A NEW PROGRAM)      !
!                                        !
!     E - EDIT/RUN (AN EXISTING PROGRAM) !
!                                        !
!     P - PRINT (AN EXISTING PROGRAM)    !
!                                        !
!     D - DISK UTILITIES                 !
!                                        !
!     M - MULTI-IMAGE UTILITIES          !
!                                        !
!     Q - QUIT THIS SESSION              !
```

C. In Case Of Difficulty

If you have any difficulty with the boot, try opening and closing the disk drive door several times to reseat the diskette. If that fails, turn off the computer, and open the top lid. Gently rock each interface card in its socket, to insure good contact. This will clear up most problems. If you still can't boot the system, contact your dealer for help.

A side note on Apple][reliability might be helpful here. Above, we briefly discussed the most frequent (not very frequent, at that!) problem with the Apple: Occasionally, the cards will not make good contact. This is true on any computer. Another connector you should know about is the cable which connects the keyboard to the main p.c. board inside the Apple (called the Motherboard). If you have any trouble with your keyboard, try rocking this connector in its socket also. Sometimes, the cards or keyboard connector must be unplugged, then replugged to clear the problem.

Other than this relatively minor problem, you should have a long, happy relationship with your Apple computer. Clear Light selected the Apple because of its reliability, good construction, switching power supply, light weight, and excellent support in the marketplace for both software and hardware add-ons. It wasn't because of all the excellent high-resolution color games with sound effects.....

You are now ready to explore the world of AMPL/M! But, before we start, lets do a very important function.

D. Backing Up The System

One of the advantages of the AMPL/M system is that it allows you to make up your own disks from your AMPL/M System Master. PLEASE! Use only the BEST quality diskettes. They only cost a few cents more, but your time is worth it, don't you think? A cheap diskette isn't worth the vinyl its made of!

Select the Disk Utilities option from the main menu by pressing <D>. You will be presented with the DISK UTILITIES MENU. This menu is described in detail in section XII of this manual. Now select the BACKUP option by pressing <B>. The screen will change again, and you will be asked for original slot/drive numbers and duplicate slot/drive numbers.

If you have a 2 disk system, simply press <RETURN> four times to accept the default values. This sets up disk drive #1 as the original or source drive, and drive #2 as the duplicate or destination drive (copy from drive #1 to #2). It is a good idea to ALWAYS use the defaults, and to NEVER copy from drive #2 to drive #1. The reason for this should be obvious: It is very easy to put the wrong disk in each drive, especially if you do it backwards from the normal way every so often...

If you have a one disk system, press <RETURN> three times, then type a <1>. This selects disk drive #1 as the original as well as the duplicate disk drive. In this case, you will be required to "swap" the original and duplicate diskettes several times in the single drive. This is a tedious process, but be careful not to mix up the diskettes during the copy process, or you will destroy one or both diskettes!

The disk slot number provides a means to address up to 6 disk drives on the SUPERSTAR. The use of the slot number will be described in section XII of this manual.

Once you have selected the disk drives for the original and duplicate diskettes, the SUPERSTAR will prompt "ENTIRE DISK OR DATA ONLY (E/D)?". This gives you the option to generate a AMPL/M "data disk", which will be described in more detail in section XII. For now, press <E>. You will now be prompted to put the diskettes in the selected drives (on a one disk system, you will be asked to put the duplicate disk in drive #1). Follow the directions, and press <RETURN> when you are ready to begin copying.

If the duplicate diskette is not blank, you will be asked if you want to erase what is on the disk. Press <Y> to continue the copy. If you get any strange error messages, see section XII for directions on how to proceed. Otherwise, you will be presented with a brand new copy of the AMPL/M system in about a minute!

Note that it is especially important to correctly seat the duplicate diskette in the disk drive. It is entirely possible to go through the entire copy process, and not be able to use the duplicate diskette! This is because it was incorrectly seated in the drive, and when removed and reinserted, it no longer is readable. Whenever you make copies, make sure you open and close the door several times after inserting the diskettes. When the copy process is completed, remove

the duplicate disk from drive #2, reinsert it into drive #1, and try to catalog it. If you can't, then you will have to copy it again.

While you're at it, make up at least one more copy. Label one of them "AMPL/M WORK MASTER", and the rest "AMPL/M WORK DISK". Don't forget to include the Clear Light copyright notice (COPYRIGHT (C) 1981 CLEAR LIGHT INC.) on each disk label on which a copy of the AMPL/M system is copied. This is required per your software license agreement, which you signed to get this software package.

Put your System Master away in a safe place, never to be used again except in an emergency. Use the "WORK MASTER" to make additional copies of work disks. It would be a good idea to place a "write protect" tab over the WORK MASTER write protect notch. These tabs are provided with the blank diskettes, and prevent the computer from writing on the disk.

The next step is to set up your work disk backup system. We recommend keeping at least two generations of disk backup at all times. The system we use at Clear Light is as follows:

1. Use some standard stick-on labels, and label three disk holders "MASTER", "MOST RECENT BACKUP", and "LEAST RECENT BACKUP". Place these labels on the holder so they are visible even when the disk is in the holder.

2. Place a work disk in the master holder, and two blank disks in the backup holders.

3. Label the three disks, using the labels provided, as shown in the example below. The program name being developed in this example is "NAVA DEMO 1982":

```
MASTER:   ------------------------------------------
          !AMPL/M WORK DISK                        !
          !                                        !
          !COPYRIGHT (C) 1981 CLEAR LIGHT INC.     !
          !                                        !
          !NAVA 1982 DEMO                  MASTER! 
          ------------------------------------------

          ------------------------------------------
BLANK1:   !AMPL/M                               #1!
          !                                        !
          !COPYRIGHT (C) 1981 CLEAR LIGHT INC.     !
          !                                        !
          !NAVA 1982 DEMO                  BACKUP! 
          ------------------------------------------

          ------------------------------------------
BLANK2:   !AMPL/M                               #2!
          !                                        !
          !COPYRIGHT (C) 1981 CLEAR LIGHT INC.     !
          !                                        !
          !NAVA 1982 DEMO                  BACKUP! 
          ------------------------------------------
```

Remember that the copyright notice is required if the backup diskettes contain the AMPL/M system. If you use the "data only" option in the backup program, no copyright notice is required on the backup diskettes. If the AMPL/M system is not on the backup disks, they can not be used to "boot" the system, but can be used to load and save AMPL/M programs that you create.

3. Use the work master disk to do your programming and editing. Backup always to your least recent backup disk, and place it in the most recent backup holder. Move the disk that was in the most recent backup holder to the least recent holder. Every time you do a backup, the backup disks will be swapped between the holders.

If you follow this system, you will seldom, if ever, lose any significant amount of work. You will always have your most recent work on the work master disk, and two generations of backup on the backup diskettes. Plan to backup your current work diskette daily, for maximum safety.

E. Running The Demo

You are now ready to run the small demo program that was included on your System Master disk. Return to the main menu by pressing the <ESC> key. Select the "EDIT/RUN (AN EXISTING PROGRAM)" option by typing <E>. The computer will ask you for a filename, so type <DEMO>, and <RETURN> (If you make a typing error, use the "back arrow" key to edit). The disk will whirr, the demo program will load, and you will be presented with the following display:

```
==========================================
!1>  2>  3>  4>  5>  6>  7>  8>  9>  0>  !
!ABC ABC ABC ABC ABC ABC ABC ABC ABC ABC !
!^---^---^---^---^---^---^---^---^---^---!
!   1>NOTE: AMPL/M DEMO PROGRAM           !
!   2 :                                   !
!   3 :                                   !
!   4 :                                   !
!   5 :                                   !
!   6 :                                   !
!   7 :                                   !
!   8 :                                   !
!   9 :                                   !
!  10 :                                   !
!  11 :                                   !
!  12 :                                   !
!  13 :                                   !
!  14 :                                   !
!  15 :                                   !
!  16 :                                   !
!  17 :                                   !
!---------------------------------------!
!AMPL/M EDITOR     HH:MM AM      MM/DD/YY!
!                                         !
!COMMAND?#                     <?> FOR MENU!
==========================================
```

The number sign (#) is often used in this manual to indicate the blinking cursor. The cursor is used to attract your attention to the place on the screen where your response is needed. Here, you see the request for a command. This is the COMMAND MODE of the AMPL/M editor.

Notice the 17 lines of program. On line 1 is the CURRENT LINE POINTER, a right angle bracket (>). This pointer tells you what "cue" you are at. Except at the beginning and end of your program, the current line will be centered on-screen, so you can see 8 cues before and after the current line.

The screen is broken into three WINDOWS. The upper window, called the STATUS WINDOW, shows the dissolve status in real time as the program is running. The center window, called the PROGRAM WINDOW, shows the current section of program. The bottom window, called the MODE WINDOW, shows the current operating mode of the editor. In this case, the editor is in the command mode.

Take a sneak preview at the command mode menu. Simply press the <HELP> key, which is also the ? key. Instantly, you will see a list of all the editor commands available in the command mode.

Press the <HELP> key again, to return to the normal editor display. Press <R> to put the editor in RUN mode. The date and menu prompt will be instantly replaced with clock time and mode, in this case, 0:00:00.00 and AUTOMATIC, as shown below:

Mode window, run off:

```
!-----------------------------------------!
!AMPL/M EDITOR     HH:MM AM      MM/DD/YY!
!                                         !
!COMMAND?#                     <?> FOR MENU!
!=========================================!
```

Mode window, run on:

```
!-----------------------------------------!
!AMPL/M EDITOR     HH:MM AM    0:00:00.00!
!                                         !
!COMMAND?#                       AUTOMATIC!
!=========================================!
```

We'll talk about this more later. For now, just press the <CUE> key. This will start the demo program running. The "COMMAND?" prompt in the mode window will be immediately replaced with "<ESC> TO STOP", and the clock display will begin counting elapsed time. If you have set up your dissolves and projectors, they will spring into action, and the status display will show dissolve status in real time. More on the status display later.

When the demo finishes, the "COMMAND?" prompt will return, and the clock will stop. Press <TB> on your keyboard. This stands for TAB-TO-BEGINNING, as you may remember from your first look at the command menu. This command will bring you back to cue 1, and disconnect the projectors (take you out of run mode).

F. Typing In Your First Program

We will describe in great detail in section IX of this manual how to enter program lines into the system. For now, just type in exactly what is shown below. If you make a mistake on a line, try the back-arrow to correct it. If the computer beeps at you when you press it, then press <CTRL-X>. This is done by pressing the <CTRL> key, holding it down, and then pressing the <X> key. This will clear (x-out) the entire line, for you to start again.

To start, press <R> to enter the run mode. The projectors will automatically home to tray position 1. Then press <I> to enter the INSERT MODE. You will see a blank line appear at cue 1, with the blinking cursor. In the mode window, where the COMMAND? prompt used to be, it now says "INSERT (REL)". This tells you that you are in the insert mode, with relative access input. More on access later. For now, take a peek at the insert menu by pressing the <HELP> key. Most of the commands you see are pretty obvious, but for now, just look at them, and then press the <HELP> key again. Enter the following lines (We will use the up-caret (^) to indicate <RETURN> or <SPACE>):

YOU TYPE IN	THE COMPUTER TYPES
C U 1 2 3 ^	CUT 1,2,3
W A . 5 ^	WAIT 0.50
1 S 1 ^	1-SEC 1
W A . 2 5 ^	WAIT 0.25
1 S 2 ^	1-SEC 2
W A . 2 5 ^	WAIT 0.25
1 S 3 ^	1-SEC 3
W A 2 . 5 ^	WAIT 2.50
S C C 1 2 3 ^	SOFT-CUT CUR 1,2,3

You noticed how the computer assisted you in typing in the lines. This is COMPUTER ASSISTED PROGRAMMING. The AMPL/M system uses this concept extensively, to minimize the typing required to enter and edit programs.

If you had your projectors connected, they executed the commands as you pressed <CUE>. In any case, the status display showed the proper status. Now press <ESC> to exit the insert mode, and return to the command mode. Press <R> to turn off the run mode and disconnect the projectors (all lamps off). You are now ready to begin learning the editor commands.

G. Control Reset and Other Problems

As mentioned earlier in this section, if you press <CTRL-RESET>, a special "interrupt" program is executed. There are several other possible system problems, usually associated with the Interface card, which can also run the interrupt program. When this program is run, the cause for the interrupt is given as a number. If you need to call the factory for help because of some unusual problem, please remember the number given when the interrupt program is called! This number is useful if some unknown problem is causing the system to crash, but normally you will only see the number 04, which indicates a CTRL-RESET. At this point, you are given two choices:

1. Press <B> to reboot the AMPL/M system.

2. Press <S> to save the current program in memory onto disk.

More information on saving programs to disk is given in section XII of this manual.

The purpose of this interrupt program is to protect your work, in case something goes wrong with the software or hardware. While it is possible that this interrupt program will suddenly run all by itself (caused by a malfunction of the Interface card or the computer itself), its main purpose is to allow you, the user, to recover from some software bug that may inadvertently appear. If you can't recover control of the computer by using <ESC>, then your LAST option is to press <CTRL-RESET>, forcing the interrupt program to run. Now you can save your current program on the disk, if desired, and reboot the AMPL/M system. If the problem persists, turn off the computer, and try reseating the cards in their sockets inside the computer.

SECTION III

AMPL/M EDITOR OVERVIEW

A. Editor Operating Modes

The AMPL/M editor has 7 distinct sections, or modes. You have already briefly seen the COMMAND MODE, the LOAD MODE, the INSERT MODE, the MENUS MODE, and the RUN MODE. The other modes you haven't seen yet are the EDIT MODE, and the QUIT/SAVE MODE.

Each of the modes are designed for a specific purpose, and are quite easy to get to from any other mode. A brief description of each mode is given below. More information is given in later sections of this manual.

1. LOAD MODE: This mode is entered automatically when you enter the editor with the E option from the main menu. You will be asked for the name of the file you want to edit. If a program already resides in the computer memory, it will first ask you if you want to edit it. If not, you will be asked if you want to save it to disk, unless a copy already exists on the disk. Then the specified program file will be loaded from disk, and control will be transferred to the editor command mode.

2. COMMAND MODE: This is a "line oriented" mode, which allows you to move about in your program, or to manipulate your program by single lines or blocks of lines. You can also go to any of the other editor modes directly from the command mode. This is the "heart" of the editor. Most commands in this mode are one or two keystrokes, and are usually executed IMMEDIATELY, with no <RETURN>. The exceptions to this rule will be described as each command is defined in section IV, VII, and VIII of this manual.

3. RUN MODE: This mode can be turned on or off in the command mode, using the <R> command. In addition, the run mode is turned off whenever the user moves to a new cue without using the <CUE> switch. When the run mode is turned on, the projectors will be synchronized to the current program cue. When the run mode is turned off, the projectors will be disconnected, with all lamps off. The run mode can be active in the command mode, the insert mode, and the edit mode. When active, program statements are executed and transmitted to the dissolves, and dissolve status is displayed in the status window.

4. INSERT MODE: This mode allows the user to type in a series of program lines, with the run mode turned on or off. Insert mode is entered by typing <I> from the editor command mode. The editor will insert a blank line at the current line pointer (>), and will place the blinking cursor at the beginning of the blank line, waiting for input. As each line is entered, the SUPERSTAR checks it for proper syntax, and enters it into the program memory when <RETURN> or <SPACE> is pressed. If run is on, the editor then executes

the program statement. A new blank line is inserted immediately following the last insert, and the blinking cursor again waits for input.

In addition to entering lines, the insert mode also allows the user to move about in his program, using various CONTROL keys. Lines may be inserted or deleted as well. The <ESC> key is used to return to the command mode, and <HELP> brings up the edit/insert menu, which displays the control commands available in the insert mode.

5. EDIT MODE: This mode is almost exactly the same as the edit mode, except that no blank line is automatically inserted when the mode is entered, and no new blank line is automatically added after a new line is typed in. The edit mode is accessed by typing <E> from the command mode. The purpose for this mode is to allow editing of existing lines, with occasional inserts and deletes, without the bother of blank lines being inserted on you automatically where you don't want them. <ESC> exits to the command mode, and <HELP> brings up the edit/insert menu.

6. MENU MODE: This mode displays appropriate menus for the new user. Most menus are accessed with the HELP key. Other menus automatically appear as required.

7. THE QUIT/SAVE MODE: Control is transferred to this mode when you <ESC> from the editor command mode. This mode allows you to either quit from the editor, or to save your program to disk. You should usually save your work every half an hour (even more often if the power in your location has a habit of going off frequently). Once you have saved your program, you may reenter the editor from the main menu.

B. Automatic Protection Features

There is no need to worry about losing your program by typing in an incorrect command. The AMPL/M system was designed to be goof proof. Any time you ask the computer to do something which would destroy your program in memory, it will always ask for a confirming YES or NO. This protects you, and allows you to operate without fear of the keyboard. The computer even knows if you have modified your program since the last disk save, and quit the editor without saving the program to disk! Thus, if you quit the editor with two <ESC>'s, and found yourself in the main menu, and then the telephone rang, when you come back later to exit the system (to go to lunch, for example), the computer will ask you if you want to save your current program. To make use of this wonderful feature, make sure you quit the system by using the Q option on the main menu, rather than just turning off the computer!

The AMPL/M editor also provides an automatic file backup option, which allows you to save your program at different stages during its development, automatically retaining two generations of backup files. This allows you to go back and see what you did on an earlier version of your work, just in case you deleted something by mistake, or you

think that a previous attempt at a sequence was better then the latest attempt. This advanced option is described in detail in section XII of this manual.

C. Editor Commands

Most of the functions performed by the AMPL/M editor are from the editor command mode. Most of the commands are shown on the command mode menu, which is displayed when the <HELP> key is pressed from the command mode. This menu is shown below:

```
==========================================
!                                        !
!AMPL/M EDITOR COMMAND MENU --------------!
!                                        !
! A  AMPL/M STATEMENTS   G# GO TO LINE # !
! BD BLOCK DELETE        GN GO TO NAME   !
! BL BLOCK LOCK          I  INSERT LINES !
! BP BLOCK PREVIOUS      M  MARK LINE    !
!#BR BLOCK REPLICATE   #PF PAGE FORWARD !
! BU BLOCK UNLOCK       #PR PAGE REVERSE !
! CA CLOCK AUTOMATIC     R  RUN ON/OFF   !
! CP CLOCK PAUSE/STEP    TB TAB TO BEGIN !
! CS CLOCK SYNCROLINK    TE TAB TO END   !
! E  EDIT LINES         #TF TAB FORWARD  !
! FR FILE READ          #TR TAB REVERSE  !
! FW FILE WRITE          X  STATUS ON/OFF!
! G  GO TO MARK                          !
!                                        !
!          CTRL-X  DELETE LINES          !
!           <--    LINE REVERSE          !
!           -->    LINE FORWARD          !
!                                        !
!          <ESC> TO QUIT EDITOR          !
!                                        !
!   <?> RETURNS TO EDITOR COMMAND MODE   !
==========================================
```

The menu shows all of the commands described in the next few sections, except the insert/edit mode compatible control commands. The number sign (#) preceding some of the commands indicates a numeric "prefix" is allowed on those commands. For example, typing <10PF> will result in the PAGE FORWARD command being executed 10 times.

Only two commands will work directly from the command menu, <ESC> and <?>. The other commands will work only from the command mode, when the COMMAND? prompt appears at the bottom left corner of the mode window. If you try to use one of these commands when in the menu, the SUPERSTAR will issue that wonderful "beep" you will grow to love, indicating an "entry" error.

SECTION IV

GETTING AROUND INSIDE YOUR PROGRAM

This section of the manual describes the many different ways in which you can move around inside your program in the command mode. You will find that there are usually several different ways of doing each move. This redundancy is designed to fit the many different situations, and the many different typing skill levels of the SUPERSTAR users. Become familiar with all the moves, before you choose the ones that best suit you.

A. Line Commands

There are two sets of line forward/reverse commands. One set is the left and right arrow keys on the right side of the keyboard. These keys will move you one line (or "cue") forward or reverse in your program. Because of their close proximity to the <REPT> key, you can easily move forward or backward at 10 lines per second. Just press and hold either arrow key, and then press the <REPT> key. Your program will scroll forward or reverse at 10 lines per second. Just release the <REPT> key to stop.

The second set of line forward/reverse keys are control keys. This means you must press two keys at once: first <CTRL>, and then the <P> or <N> key (<CTRL-P> or <CTRL-N>). The "P" stands for "previous line", and is equivalent to the left arrow, and "N" stands for "next line", and is equivalent to the right arrow.

The purpose for the control key set of commands is compatibility with the movement commands in the insert and edit modes. The control keys will work in all three modes, while the arrow keys will work only in command mode. The arrow keys are easy to use to "browse" around in your program, especially if you are not a typist, while the control keys are easiest to use if you type, or are switching back and forth between the edit/insert modes and the command mode.

B. Page Commands

There are also two sets of page forward/reverse commands. The first set is available only in the command mode. They are <PF> for "page forward", and <PR> for "page reverse". When these commands are executed, the program quickly (20 lines per second) scrolls 15 lines in the indicated direction. If you use a number prefix, such as <10PF>, the SUPERSTAR will scroll forward 10 pages, stopping momentarily at each page. This allows you to quickly scan your program for a particular sequence. If you want to stop, simply press <SPACE> or <ESC>.

The other set of page commands are control commands, again compatible with the insert and edit mode control commands. They are <CTRL-F> and <CTRL-R>, for "page forward" and "page reverse". Both these commands may be used with a number prefix, but only in the command mode.

The page commands also have a speed control available. When a page command is being executed, you may press any single digit from 0 to 9 to select a slow or fast scroll rate. The 0 rate is actually a page "flip", which means the page is changed instantaniously. Speed 1 is the fastest, and speed 9 is the slowest. The value is initialized to 2 when the system is booted, and will remain at the selected speed until you change it. Select the speed that you feel most comfortable with!

C. Goto Commands

The GOTO commands allow you to move instantly to any cue in your program. This is a fantastic time saver, assuming you know what line number you want to go to! AMPL/M provides a solution to that, too. More on that shortly.

To move to a specific line number, press <G> for "goto", followed by a 1 to 4 digit line number. If you make a mistake in the line number, use the back arrow to edit. A four digit line number will enter automatically. For a shorter number, press <SPACE> or <RETURN> to execute the goto. Presto! You're at the desired line number! A <G1> will go to the beginning of your program, and a <G9999> will always go to the end.

Now the goto line number is great, but not if you can't remember what the cue number is that you want to go to. Enter the powerful AMPL/M name concept. You may give names to particular cues in your program, using the TAB (or TASK) statement. For example:

```
TAB: CREDITS
1-SEC 1,2,3
WAIT 2.50
2-SEC NXT 1,2,3
WAIT 3.65
CUT 2,4
WAIT 0.50
4-SEC CUR,THD 1,2,3
WAIT 0.24
1-SEC CUR 2,4
WAIT 4.50
TAB: SUNRISE
REPEAT 24
 :
 :
 ETC.
```

In the example program above, the credits sequence is named "CREDITS", and the next sequence is named "SUNRISE". Each sequence in your program may be so named. An AMPL/M name may be up to 18 characters long, must start with an alphabetic character, and cannot contain any spaces. Any printable character is allowed after the first character. The hyphen (-) and the underscore (_) may be used as "spaces" if desired.

To go to a named sequence, simply type in <G> followed by the name. For example, to go to the SUNRISE sequence, type <GSUNRISE>. Press <RETURN> or <SPACE>. Presto! You are at the desired sequence!

If you misspell the name, or if the computer can't find the name you requested, it will simply beep at you, and not move at all.

D. Tab Commands

There are other ways to use the tabs discussed above. Often, you will find yourself going back again and again to the same point in your program, to try a sequence until you get it timed right (using SYNCROLINK, discussed in section VI of this manual). Wouldn't it be a drag if you had to type in the name every time? Well, we thought so! Simply use the tab commands, which allow you to move forward or backward in your program one or more tabs, regardless of name. The <TF> command moves forward ("tab forward"), and the <TR> command moves back ("tab reverse"). TASK statements operate like a tab, as well. Also, the number prefix is allowed on these commands. For example, typing <4TR> will transport you four tabs and/or tasks backwards in your program.

In addition to the tab forward/reverse commands, the AMPL/M editor allows two other handy tab commands: <TB> for "tab to beginning", and <TE> for "tab to end". These two commands provide another simple way to get to either end of your program quickly.

When you tab to the end of your program, you will find the current line pointer is pointing at a blank line AFTER the last cue in the program. This is the END-OF-PROGRAM line. This line is necessary in order to allow you to append lines to the end of your program. Insert always makes a blank line in front of the current line, and after the previous line, so without the end-of-program line, it would be impossible to insert AFTER the last program line! To append lines to the end of your program, simply type <TEI>. This will tab to the end of the program, and enter the insert mode after the last line of your program.

E. The Cue Key

The SUPERSTAR computer uses the <RETURN> key as the <CUE> key as well. In some cases, this may be confusing. We have designed the commands for AMPL/M to minimize this problem.

The <CUE> key will act as a line forward command if the run mode is off. If the run mode is on, then the <CUE> key also triggers a multi-image "cue" to the system clock.

In addition to the <RETURN> key, the SUPERSTAR allows a remote switch to trigger a cue as well. The remote cue works exactly the same as the <RETURN> key, but it can be enabled or disabled using the AMPL/M program statement "REMOTE". This is described in appendix I. The remote cue is automatically enabled when the editor run mode is invoked, and must be turned off by the program if not wanted.

A third source of a cue is via an "EXECUTE" cue received at the sync-in jack on the rear panel. Again, this cue works exactly the same as the <RETURN> key. The computer literally doesn't care where the cue comes from! The execute cues on tape are used to automatically start up the execution of cues when doing "tape

activated syncrolink", described in section XVI of this manual.

A fourth source of a cue is from a clock track. The clock track, under the right conditions, can automatically cue the program to begin, or resynchronize the program to a different cue. This is described in detail in section XV of this manual.

SECTION V

RUNNING YOUR PROGRAM

This section of the manual deals with the use of the run mode of the AMPL/M editor. This mode is used not only to run a finished show from the SUPERSTAR, but also to refine and test the show during the programming phase.

A. Autosync

Those users of the Clear Light Star System are familiar with the autosync function, because it works exactly the same on the SUPERSTAR as it does on the Star-3 system. Rather than requiring you to wait for the projectors to follow you anywhere you go in your program, or losing track of where the dissolves are if you don't, the SUPERSTAR allows you to move with or without the projectors following. It's your option! In addition, the computer WILL NOT FORGET where the dissolves and projector trays are, no matter what you do (unlike some other systems on the market!). The operational rule is very simple. If you move to some other cue in your program with a move command, the projectors will "disconnect", meaning they will shut off to standby mode. To bring them along with you, simply enter run mode. If you are in command mode, this is a single key stroke, <R>. If you are in the insert or edit mode, simply type <ESC><R>. The computer will calculate the appropriate moves for the projectors, and bring them to the correct position!

It is very easy to tell what mode you are in by looking at the mode window of the editor. Below is a diagram of the mode window (with fake time and date) with run mode on and off:

Mode window with run mode OFF:

```
!-----------------------------------------!
!AMPL/M EDITOR     11:15 AM       2/24/82!
!                                         !
!COMMAND?#                    <?> FOR MENU!
!=========================================!
```

Mode window with run mode ON:

```
!-----------------------------------------!
!AMPL/M EDITOR     11:15 AM    0:00:00.00!
!                                         !
!COMMAND?#                        AUTOMATIC!
!=========================================!
```

The runtime clock value is shown in the upper right of the window, and will be zero whenever autosyncing to cue 1 of the program. The prompt "AUTOMATIC" tells you the clock mode. More on that shortly. Note that you are still in command mode at this point. The show is not running, and the system clock is not counting. Don't confuse the system clock with the time-of-day clock, which is always running.

If you use the <CUE> switch to move from the current cue to the next cue, the system clock will be started, and the program will begin to execute. This can be seen in the mode window by the clock value counting up, as well as the disappearance of the "COMMAND?" prompt, as shown below:

Mode window with run mode ON, and the system clock running:

```
!-----------------------------------------!
!AMPL/M EDITOR      11:15 AM     0:00:00.00!
!                                         !
!<ESC> TO STOP                   AUTOMATIC!
!=========================================!
```

The "<ESC> TO STOP" message tells you how to stop the clock, and return to the command mode. This prompt will remain until you hit <ESC>, run out of cues, or hit a "cue request" in your program. Cue requests are described in the next section of this manual. If you press <ESC>, or the end of program is reached, the "COMMAND?" prompt will reappear in the mode window, signifying that you have returned to command mode.

B. Dissolve Status

As mentioned before, the top window of the editor display shows the dissolve status in real time. Each dissolve has a little 3 x 3 box, containing all the information about its status. The meaning of each item in the box is described below:

1. DISSOLVE NUMBER: The single digit at the top left of each box is the dissolve or screen area number. AMPL/M presently supports up to 10 Star-3 Dissolves, for 30 projector operation. The screen numbers go from 1 to 9, and 0 for 10. This number will normally be in normal video (white on black). However, if a dissolve is put into the Absolute-Access mode, the screen number will appear in reverse video (black on white). The operation of the different dissolve modes is covered in detail in section XIX.

2. ADVANCE MODE: The Star-3 Dissolve supports three different advance modes: Normal advance, No-delay, and No-advance. The normal mode is indicated by one right angle bracket (>) after the screen number. This symbolically indicates that you are "moving forward" in your show, because the projectors are advancing after each dissolve. The no-delay mode is indicated by two right angle brackets (>>), because you are moving "faster" through your slides. Naturally, the no-advance mode is indicated by the absence of any right angle brackets. For more details on the Star-3 Dissolve modes, see section XIX.

3. PROJECTOR LAMPS: The second line of the status box contains the projector letters "ABC", referring to the projectors connected to the three cables on the rear of the Star-3 Dissolve. The projector letters are in normal video if the projector lamp is off or fading down, and are in reverse

video if the lamp is on or fading up.

If the dissolve is put into two projector mode (see the freeze command description in section XIX), then only the A and B letters will appear.

4. PROJECTOR ACCESS: The bottom line of the status box tells you something about the automatic sequencer built into the dissolve. The little up-caret (^) points to the "next" projector in the automatic sequence. A hyphen (-) is used to indicate an available projector for automatic or program dependent access, and a plus (+) is used to indicate a projector in freeze mode. Any projector in freeze mode may not be accessed except with Absolute-Access. For more details on these various modes, see section XIX of this manual.

C. Tray And Program Status

There is even more status information provided to you by the AMPL/M system. From the editor command mode, type <X> to see the extended status page.

The extended status page shows you not only tray position for each of the 30 projectors, but also other important status about your program. Each item is described below:

1. TRAY STATUS: This shows the current tray positions of the slide projectors, if run mode is active. The system assumes that the projectors start off in tray position 1. This seems to be the "standard" position to which slide trays are set at the beginning of a show.

2. COMMAND COUNT: This figure tells you how many lines you have in your program now residing in memory.

3. COMMANDS FREE: This figure tells you how many more cues will fit in memory. This figure assumes no notes, tabs, or tasks will be used (notes, tabs, and tasks take additional space beyond a normal cue).

4. NOTES USED: This is a character count for all the notes and names used in the program. As you enter programs, both notes/names and cues take up memory space. If you have a long program that doesn't quite fit in memory, and you don't want to load another file during the show, than you can free up more memory space by deleting or shortening notes and/or names.

5. DELETED NOTES: This number represents all the characters which were in note and name statements which have been deleted from your program. This is "wasted" memory space. If you need to get this space back, simply quit the editor, and save your program to disk. Then reenter the editor, and load the file back off disk (answer the "EDIT CURRENT PROGRAM (Y/N)?" question with a <N>). When the program is saved to disk, the wasted space is eliminated, and when

reloaded, is returned to free space. You can usually expect to recover one cue for every four characters returned to free space. This process of recovering wasted memory space is called GARBAGE COLLECTION in computer lingo.

D. Program Speed

The SUPERSTAR computer has the capability to run cues in real time at 100 cues/sec. However, depending on what TARGET you will play back the show on (what set of equipment you will use in show playback), you may wish to restrict the computer to a slower speed.

The AMPL/M language has a speed statement for this purpose. The statement allows the user to set the speed to one of three values: 100 cues/sec, 50 cues/sec, or 10 cues/sec.

For compatibility purposes, the SUPERSTAR assumes the lowest speed if not specified otherwise. If you intend to run the program directly from the SUPERSTAR, or from a Star Memory, than use the "SPEED: 100 CUES/SEC" statement at the beginning of your program. If you are using the Star Universal Interface to play back your show, use the "SPEED: 50 CUES/SEC" statement. If you are going to use the Star-3 Dissolves directly to play back your show, then use the "SPEED: 10 CUES/SEC" statement. While it isn't necessary to use the 10/sec statement (because the computer assumes it if you don't say otherwise), we recommend you use it anyway, for clarity.

The SUPERSTAR will limit its rate of cue production to the specified speed. If you specify a faster cue rate in a wait statement in your program, it will delay the cue the appropriate amount of time, and give you a warning message. If it gets too far behind, because the speed of the sequence is much faster than the specified speed, the computer will stop, and give an error message.

Keep in mind that the full 1/100 second resolution is still available to you, even at the slowest cue speed! The cue speed sets only the MINIMUM separation between cues.

Appendix I describes how to enter the speed statement into your program.

E. The Runtime Clock

Built into your SUPERSTAR (on the Clear Light Interface Card) is a precision chronometer, which regulates the timing of your show as it runs. This clock keeps track of how many hours, minutes, seconds, and hundredths of a second have elapsed since your show began. The time data is displayed in the format H:MM:SS.FF on the upper right of the mode window when in run mode. The maximum length of time in a single show is 2 hours, 39 minutes, 59 seconds, 99 hundredths.

All system timing is based off this system clock. The absolute time statement, WAIT UNTIL T, uses this elapsed time clock. The relative time statements, WAIT N or WAIT X, are timed off this clock as well. All relative time statements have a minimum time of 0.01 seconds and a maximum time of 99.99 seconds.

The clock can be operated in one of three modes, described below. The active clock mode is always displayed in the bottom right corner of the mode window, whenever the run mode is active.

1. AUTOMATIC: This clock mode is used to run the show in full real time, and is automatically selected when the editor is initialized. The user can return to this mode from one of the other clock modes by typing <CA> when in the editor command mode. The system clock will automatically lock to any incoming clock track, or will run independently if none is present.

   The clock is started by entering the run mode, and then cueing the computer. The cue can come from the <RETURN> key, the remote cue switch, an execute cue off tape, or a clock track. If the editor is in edit or insert mode, the clock will run long enough to execute the current line. Otherwise, the clock will run until stopped by the user with the <ESC> key, until the maximum clock time is reached, or until no more program statements are available.

   Note that just because the clock is running does not mean that the show is progressing! For example, the system may be waiting for a manual cue to proceed. However, the clock is still running, because time is elapsing (projector lamps are burning, etc.).

2. PAUSE/STEP: This mode is identical to the Star-3 System Autopause/Single-Step, and is selected by typing <CP> in the command mode. The clock runs all waits, but when the next dissolve statement is reached, it freezes the system. All lamps stop at whatever intensity they are at. When <CUE> is pressed, the next statement is executed, including any waits, until the next (non-wait) statement.

   Basically, what happens on the screen is the projected images retain their exact illumination relationships, even though you are single-stepping through your program. This advanced feature allows you to debug your high speed effects easily. This clock mode can also be used in the insert or edit modes, allowing you to enter lines in pseudo real time. This means that the program will execute in steps as you enter them, and not lose projector lamp relationships because of the statement entry time.

3. SYNCROLINK: This is probably one of the most important features of the Star System. It is activated by typing <CS> from the editor command mode. It allows the user to automatically generate exact timing for his program. Don't confuse this with the WAIT X or syncrolink on other systems! The SUPERSTAR version of syncrolink is a true computer assisted programming function that will save literally hours of programming time! The syncrolink feature is covered in sections VI, VIII, XVI, and XX of this manual.

   The syncrolink clock will automatically lock onto an incoming clock track. Otherwise, the internal chronometer

will be used for timing.

SECTION VI

SYNCROLINK AND TIMING: AN INTRODUCTION

The concept of SYNCROLINK was introduced to the multi-image community in 1978 with the introduction of the Star-3 System. The basic idea is so simple, it is amazing that it wasn't discovered long before. In fact, we had difficulty explaining this feature to many people, not because they couldn't understand what we were telling them, but because they couldn't believe it!

Syncrolink simply allows the user to automatically enter his program timing by cueing the program in real time, as he listens to the sound track, rather than requiring him to calculate the timing laboriously by hand with a stopwatch, and enter it manually. In addition, the user could repeat the timing exercise over and over until he got it right. If necessary, subsections of the sequence could be retimed, without changing the timing of the remainder of the sequence.

AMPL/M extends the concept of syncrolink to new areas, which make the timing and perfecting of a show even easier and less painful than even the Star-3 System!

A. Waits And Cue Requests

The AMPL/M language has been patterned after the Star-3 System in many obvious ways. However, at first glance, you might think that the timing statements on the SUPERSTAR, such as WAIT X, are more like those found on other mulit-image programmers on the market. In fact, the timing statements of AMPL/M work in the same way as the timing "links" of the Star-3 System.

The original Star System had a timing link associated with each cue in memory. This link could be "zero", which meant that the system would stop at the next cue, and wait for a manual cue from the user (or an execute cue off tape). If the link was not zero, then the indicated time delay was executed, and the next cue was executed automatically. Thus, the timing link was a delay between the current cue and the next cue.

On the SUPERSTAR, we decided to separate the two pieces of the Star System cue, the A/V "command" and the timing link. However, the basic concept remains the same. If a zero timing "link" exists between two cues, the computer will stop for a manual cue. A zero link on the SUPERSTAR is made by not using a WAIT statement between cues. Look at the following example:

```
CUT 1
CUT 2
CUT 3
```

This sequence is just like a Star-3 sequence of three cuts with zero links. This means that the user must manually cue the sequence: no automatic timing is indicated. When the computer stops for a

manual cue, this is referred to as a "cue request". This means that the program is requesting a cue from the system clock, which received its cues from the <CUE> key, the remote cue switch, an execute cue on tape, or a clock track.

Whenever the program is requesting a cue from the clock, the "<ESC> TO STOP" prompt in the mode window is changed to "CUE?", as shown below:

Mode window with cue request pending:

```
!---------------------------------------!
!AMPL/M EDITOR    11:15 AM     0:02:15.43!
!                                       !
!CUE?#                          AUTOMATIC!
!=======================================!
```

The "CUE" prompt tells the operator that he must press <CUE> (or provide some other cue source) for the program to continue running. Once the cue request has been satisfied, the prompt returns to "<ESC> TO STOP".

Other AMPL/M statements can also make "cue requests", as well. In fact, with multiple time lines, using the TASK feature of AMPL/M, the SUPERSTAR has to deal with many potential cue requests at once! More on this shortly (The TASK operation is covered in section XX of this manual).

When you wanted a 0.1 second delay between cues in the Star-3 System, you specified 0.1 seconds. There was no "implied" execution time in the cue execution itself. Likewise for the SUPERSTAR. Look at the following example:

```
CUT 1
WAIT 0.10
CUT 2
WAIT 0.10
CUT 3
WAIT 0.10
```

In the example above, the cuts will automatically execute at ONE TENTH of a second apart, since the wait statements indicate automatic timing. On other systems, this sequence would execute at either 0.15 second intervals, or at 0.20 second intervals, depending on the "run speed". For this reason, the AMPL/M program is much more readable, because it says exactly what happens in the program! No calculations in your head. No confusion when trying to add up the total time of a sequence.

Notice that all wait statements use two digits following the decimal point. This is because all waits can be expressed in increments of 0.01 seconds, from 0.01 to 99.99! No more "aux-control-fast" statements (for you Star-3 users). The only exception to this is when programming for a Star-3 Memory playback target. This is described in section XIII.

It is very important to understand the basic concept here, before trying to use the syncrolink function. As you will begin to see, the

timing system only LOOKS similar to other systems on the market, but in reality is substantially different, and much more powerful.

B. Implied Wait For Cue

As explained in the example above, the SUPERSTAR will automatically stop if there is no wait cue between dissolve commands, and prompt for a cue in the mode window. This is referred to as an "implied WAIT FOR CUE". Rather than make you enter a WAIT FOR CUE statement every place you wanted the computer to stop, the computer automatically stops if no wait is specified.

There are several types of program statements in AMPL/M which are not commands to the dissolve, but serve important functions in the language, such as REPEAT, LET, NOTE, etc. These statements are called control statements (REPEAT, etc.), or documentation statements (NOTE). These statements do not interfere with the generation of automatic WAIT FOR CUE's. Thus, any number of control or documentation statements occurring between two dissolve statements will not change the fact that no timing link exists between the dissolve statements, and a cue request will be generated, unless at least one WAIT statement exists between the dissolve statements. Look at the example below:

```
NO-ADVANCE 1,2,3
REPEAT 25
 CUT 1
 WAIT 0.1
END: REPEAT
```

The computer will stop at the CUT 1 statement when it first encounters it, because no time was specified between the NO-ADVANCE and the CUT. A cue request will be issued, and the CUE? prompt will be displayed in the mode window. If no manual cue was desired, add a WAIT after the NO-ADVANCE, or reverse the order of the CUT and WAIT statements in the loop. If you change the order in the loop, you must follow the END: REPEAT statement with a WAIT also, otherwise the computer will stop at the next dissolve statement.

C. Wait X

The WAIT X statement is an adjustable version of the WAIT N statement described above. "Adjustable" means that the syncrolink timer can be used to set its value. The WAIT X statement can also be "initialized" to a value when it is first entered by the user into his program. For example, to enter the statement "WAIT X = 2.50", type <WX=2.5>. Details on this are given in appendix I.

When a WAIT X statement is entered into the computer, the X suddenly turns "inside out" to reverse video. This indicates that the value of X is "unlocked", and may be changed by the syncrolink timer. The value may be "locked" when the user no longer wants the value changed by syncrolink. This is explained in section VIII of this manual. When locked, the X returns to its normal video self.

When syncrolinking a WAIT X statement, the SUPERSTAR replaces the <ESC> prompt, as shown below:

Mode window with X request pending:

```
!-----------------------------------------!
!AMPL/M EDITOR    11:15 AM     0:02:15.43!
!                                         !
!CUE?# X = 00.01                AUTOMATIC!
!=========================================!
```

The X value displayed will increment in real time, showing you the current value of the syncrolink timer. When you press the <CUE>, the value is stored in the WAIT statement, and redisplayed in the program window. The "CUE" prompt is then replaced with the normal "<ESC>" prompt.

Lets look at a program segment to see how this works. Below are two programs, one before syncrolink, one after syncrolink:

before	after
CUT 1,2,3	CUT 1,2,3
WAIT X = 1.25	WAIT X = 1.43
CUT 1	CUT 1
WAIT X	WAIT X = 0.26
CUT 2	CUT 2
WAIT X	WAIT X = 0.52
CUT 3	CUT 3
WAIT 2.40	WAIT 2.40
CUT CUR 1,2,3	CUT CUR 1,2,3

The first WAIT X statement was either initialized upon entry, or was previously syncrolinked. Notice that the syncrolink did not convert the WAIT X statements into WAIT N's! This is a very important point! The statements now have a defined value, and can be used like WAIT N statements simply by running without syncrolink, but they also can be re-syncrolinked without any program changes! Once the values are satisfactory, each individual WAIT X or all the WAIT X statements in a specified section of the program can be locked with a single command. This is described in section VIII of this manual.

If the SUPERSTAR encounters a WAIT X statement with no defined value for X while running a program, and the clock is not in syncrolink mode, than the computed value of X will not be stored in the WAIT X statement when the cue is received. In this way, the WAIT X is treated functionally the same as a implied WAIT FOR CUE, because a cue is required for the program to continue. Usually, it is a good idea to initialize the values of WAIT X statements when entering the program, unless you intend to immediately syncrolink them.

D. Repeat Loops

It is also possible to syncrolink the repeat value of a loop. This requires a REPEAT C statement to begin the loop. The value of C can be initialized when the statement is first entered, just like the X value described above. Below is an example of a REPEAT C loop:

```
REPEAT C = 12
 CUT 1
 WAIT 0.10
END: REPEAT
```

The loop will automatically loop 12 times, generating 12 cuts to screen area 1 at a rate of 10 per second. Note the automatic indentation of all statements within a loop. This is done to make it easy to see the range of the loop. This is described in greater detail in section XX of this manual. Also note the use of "C" rather than "X" in the REPEAT statement. The "C" stands for "counter", and is different than an "X" time value in range and type. This distinction is important when extending the AMPL/M language to include program variables, which are discussed in section XX of this manual.

Like the X in WAIT X statements, the C in REPEAT C will be inverse video until locked. If the loop in the example above is executed with syncrolink active, the value of C will be recomputed. The REPEAT C statement will generate a cue request, which will be prompted in the mode window, as shown below:

Mode window with C request pending:

```
!-----------------------------------------!
!AMPL/M EDITOR    11:15 AM     0:02:15.43!
!                                         !
!CUE?# C = 0001                 AUTOMATIC!
!=========================================!
```

The value of C will count up how many loops have been executed so far. When the <CUE> key is pressed, the current value of the loop counter will be stored automatically in the REPEAT C statement, and the REPEAT statement will be redisplayed in the program window with the new value. The "CUE" prompt will be replaced with the normal "<ESC>" prompt.

The REPEAT C statement is similar to the REPEAT X function of other systems, except the repeat count is exactly equal to the number of repeats, not one less. The repeat statement itself is at the beginning of the loop, rather than the end, which is the standard for all other computer systems in the world (other than multi-image). Again, this makes AMPL/M programs more readable.

Now what happens if you mix the X and C types of cue requests? Lets look at an example:

```
REPEAT C
 CUT 1
 WAIT X
 CUT 2
 WAIT X
END: REPEAT
```

In this example, there are three statements which are looking for a cue; the REPEAT C, and two WAIT X's. It is very important to understand what the computer will do with these three cue requests. First of all, the WAIT X requests are "immediate". This means that they must be serviced before going on to the next statement. On the

other hand, the REPEAT C cue request is "deferred", which means it must be received after the statement has executed. When the loop is entered (assuming syncrolink is active), a deferred cue request is put in the "cue request queue", and the "CUE?# C = 0001" prompt appears in the mode window. When the computer encounters the first WAIT X, an immediate cue request is issued, and the computer waits for it before proceeding. The "C = 0001" prompt is immediately replaced with the "X = 00.00" prompt. This happens so fast in this case (no wait between the REPEAT C and first WAIT X statement) that you will not actually see the "C" prompt. When the immediate cue is received, the value of X is computed and stored in the WAIT X statement. The computer continues to the next WAIT X, and handles it in the same way. Now that all immediate requests have been met, the next cue will go to service the deferred cue request. Now you will see the "C" prompt in the mode window. When the cue is received, the value of the C counter will be stored in the REPEAT C statement, and the loop will terminate. Note that WAIT X's only are computed on the first pass through the loop, and these values will be used for all succeeding repeats.

One more example should make the operation of the cue requests and their respective prompts clear:

```
REPEAT C
 CUT 5
 WAIT 1.00
 CUT 1
 WAIT X
 CUT 2
 WAIT X
 CUT 3
 WAIT 1.00
END: REPEAT
```

This loop is similar to the previous example, except some fixed one second waits have been added. Now there IS time for the "C" cue request prompt in the mode window. During the first WAIT 1.00, the "C" request will appear in the window. If you press <CUE> at this time, the value of C in the REPEAT C statement will be set to 1. After a second, the "C" prompt will be replaced with an "X" prompt, and the loop will proceed as before. When the second WAIT 1.00 is executing, you have another chance to terminate the loop with one repeat, if you haven't done so already! Keep your eye on the prompt to see what you are cueing, and you won't get into trouble! Not only will you be sure which cue you are entering, but also, if you press the <CUE> switch when no cue request is pending, the keystroke will be ignored.

If a REPEAT C statement is encountered with no defined value, and syncrolink is not active, a "C" type cue request will be issued, but the computed value will not be stored when the cue is received.

The cue request situation becomes more complex with multiple level loops and multiple time lines. These cases are covered in section XX of this manual.

E. Until Cue

The examples above demonstrate the operation of both types of cue requests. It is possible to generate deferred requests with UNTIL statements, as well. Let's explore this possibility, using the following example:

```
REPEAT C
 CUT 1
 WAIT X
 UNTIL CUE
 CUT 2
 WAIT X
END: REPEAT
```

In this example, both the REPEAT C and the UNTIL statement will make a deferred cue request. The UNTIL CUE statement is used to allow a non-syncrolinked manual exit from either a lengthy or infinite loop. More details on the UNTIL statements will be given in section XX of this manual. For now, let's look at the cue request problem this example demonstrates.

Once the immediate cue requests are serviced, the computer is looking for two more cues. The first one received will be given to the cue request from the REPEAT C statement. This will terminate the loop, which will delete the UNTIL CUE request from the cue request queue. In this way, the UNTIL CUE statement will have no effect in syncrolink. The reason for this is the REPEAT C request overrides any UNTIL CUE requests within the same loop.

This may be surprising to you, but if you think about it, it will make sense. The purpose of syncrolink is to define adjustable values in your program. The purpose of statements like UNTIL CUE is to allow the operator to modify the program at show time. Thus, when syncrolinking the loop above, the programmer is setting the maximum length of the loop. When the loop is run without syncrolink, or when the C value is locked, the loop will run until the defined C count is reached, or until a cue is received. Amazing, but really quite logical. By the way, a loop counter can have any value from 1 to 9999 (a very, very long loop!).

F. Wait Until T

The WAIT UNTIL T statement is similar to the WAIT X statement, including the inverse video T, initial value entry, and syncrolink. The main difference is that it uses "absolute" clock time rather than "relative" clock time. The value is expressed in hours: minutes: seconds. hundredths. For example, 1 hour, 25 minutes, 39.38 seconds is expressed as "1:25:39.38". The AMPL/M clock can go up to 2 hours, 39 minutes, and 59.99 seconds.

The WAIT UNTIL T statement is used to set an absolute time reference at the beginning of sequences. This is helpful when using clock track, or resolving timing ambiguities with multiple time lines (described in section XX). You can also use this statement as a stop watch when using the clock track. Simply enter the desired number of WAIT UNTIL T statements, and syncrolink them to your clock track. You

can then read out the clock values at your leisure.

SECTION VII

THE MARK COMMAND

One of the powerful new features of the AMPL/M system is the MARK command. This command is available in the editor command mode. Simply by typing <M>, the user can mark a line for future reference. The editor displays the marked line by putting the line number in inverse video. The line will remain marked until turned off by remarking, or by being used in a block command, described in the next section. Note that marks are temporary, and are not actually part of the program. Rather, they are a "memory" function of the AMPL/M editor.

A. Goto Mark

Once the line is marked, you can move anywhere else in the program, using the movement commands described in section IV of this manual. When you want to return to the marked line, simply type <G> <RETURN> or <G> <SPACE>. Presto! You're back at the mark! Think of it like a bookmark.

The AMPL/M editor will allow up to two marks in a program at any one time. The "goto mark" command will always take you to the nearest mark. Once you are at one of the marks, another "goto mark" command will take you to the other mark. This allows you to easily check out the position of the marks in your program.

B. Marking A Block

A major new feature of the SUPERSTAR editor is its block oriented commands. These commands are made possible by the mark command, which allows you to specify a range of program lines to operate on. To mark a block, simply go to the first or last line of the program section you wish to use, and press <M>. Then move to the opposite end of the program section, and press <M> again. The two marks will delineate a "block", which can be used with the editor block commands, described in the next section. Not only will the marked lines be displayed with reverse video line numbers, but every line within the block will be marked with a colon between the line number and the program statement. This provides an absolute identification of the marked block, so no mistake is made, such as forgetting a previous mark. An example of a marked block is shown on the next page. The @ symbols are used to simulate inverse video (hard to do with a printer! If you think that's tough, try blinking characters!).

```
=============================================
!1>  2>  3>  4>  5>  6>  7>  8>  9>  0>  !
!ABC ABC ABC ABC ABC ABC ABC ABC ABC ABC !
!^---^---^---^---^---^---^---^---^---^---!
!  24 CUT 1,2,3                           !
!  25 WAIT X = 1.24                       !
!  26 REPEAT 24                           !
!  27  SOFT-CUT 1,3,5                     !
!  28  WAIT 0.15                          !
!@@29: REPEAT 3                           !
!  30:  CUT NXT 2,4                       !
!  31:  WAIT 0.10                         !
!  32>  CUT THD 2,4                       !
!  33:  UNTIL LAST                        !
!  34:  WAIT 0.10                         !
!@@35: END: REPEAT                        !
!  36  UNTIL LAST                         !
!  37 END: REPEAT                         !
!  38 NO-ADVANCE 1,2,3,4,5                !
!  39 WAIT 1.25                           !
!  40 REPEAT C = 224                      !
!-----------------------------------------!
!AMPL/M EDITOR    10:55 AM     0:00:45.81!
!                                         !
!COMMAND?#                      AUTOMATIC!
=============================================
```

Notice how easy it is to see the marked block, from cue 29 to cue 35. Also notice the centered current line pointer, which will overprint on top of a colon block indicator.

Since only two marks are allowed, if a third is made, the OLDEST mark is deleted. Also, if you mark a line which is already marked, the mark will be deleted. The mark function thus acts like a toggle function on a specific line.

There is only one line in your program which you cannot mark. This is the end-of-program line, just after the last numbered line in your program. You will find yourself on this line after a <TE> or <G9999> command. If you press <M> when on this line, the AMPL/M editor will actually mark the previous line, which is the last numbered statement in your program.

Now that you know how to mark a block, what now? The use of blocks is covered in the next section, plus in section XII.

SECTION VIII

THE BLOCK COMMANDS

The mark command is used to define a block of AMPL/M program lines (see section VII for details). This allows the user to specify an edit operation on a block of statements rather than one at a time.

Whenever a block function is requested by the user, the AMPL/M editor will prompt "NNN LINES IN BLOCK: OK? (Y/N)", where NNN is the number of lines in the block. This is insurance against performing a command on a block other than the one you had in mind. This prompt, plus the visual display of the block on the CRT screen, should keep the user from accidentally damaging the program.

Once a block definition has been "used" by a block command, the marks are removed. The last set of marks can be restored using the BLOCK PREVIOUS command, described below.

If less than two marks are present in the program when a block command is invoked, the editor will assume a block length of 1. With one mark, the marked line will be used as the block. If no marks exist, then the current line is selected as the block. It is a good idea to use one mark for a one line block command, because it will quickly reveal the unsuspected presence of previous marks still existing in the program. You can tell this by the sudden appearance of colon marks going up or down from the current line to the other unsuspected mark off-screen. Type <G><SPACE> <M> <G><SPACE> to go to the other mark, delete it, and return to the current line.

A complete description of each block command provided in the AMPL/M editor follows.

A. Block Delete

This command is entered in the editor command mode by typing <BD>. The command will remove the entire marked block of AMPL/M statements from the program. Before responding to the "OK?" prompt, make sure you really want to delete all those lines! There is no way to get them back once they are gone!

B. Block Replicate

This command is used to make a copy of a program block, and is entered by typing
. The copy is inserted in the program at the current line pointer. If there isn't enough room in memory, an error message will be given, and the command aborted, with memory full, and the block marks intact. Otherwise, the replication will be completed, and the block marks will be deleted.

To make more than one copy of a block, enter a number prefix before the
. For example, to make 10 copies of a specific block, mark the block, set the current line pointer to the desired destination, and type <10BR>.

You cannot replicate a block within itself. This means you must move the current line pointer outside the block before using the command. If you don't, the computer will beep, and ignore the command. If the current line pointer is at the first line in the marked block, the copies will be inserted immediately before the block. If the current line pointer is at the last line in the marked block, the error beep will be given, because the insert would take place before the last statement in the marked block, which is within the block.

The block replicate command has many uses, but one of the most exiting is the generation of "templates". A template is a sequence of AMPL/M statements which must be repeated several times, with one or more statements remaining the same on each repeat, but also one or more statements changing on each repeat. The block replicate command can replicate the common statements, and the editor can be used to insert the changing statements. An example of this will be given in section IX of this manual.

C. Block Lock

This command is used in conjunction with the syncrolink operation, and is entered by typing <BL>. Adjustable wait values X and T and adjustable loop counter values C are affected by the syncrolink timer, If they are unlocked (displayed in inverse video). The block lock command is used to lock the selected values, so they cannot be changed by syncrolink. When this is done, the X, T, and C characters are displayed in normal video. When the block lock is completed, any marks will be deleted.

If you wish to lock a single AMPL/M statement, simply move the current line pointer to the selected line, and type <BL>. When the SUPERSTAR responds with its "NNN" prompt, make sure it is a "1". If it isn't, there must be another block marked somewhere else in your program!

When a block lock is executed, any undefined X, C, or T values will not be locked. They must first be defined, either by initialization (when entered) or with syncrolink.

D. Block Unlock

The block unlock performs the opposite function of block lock, and is entered by typing <BU>. All WAIT X, REPEAT C, and WAIT UNTIL T statements in the block will be unlocked, and the X, C, and T characters will be displayed in inverse video. Also, any block marks will be deleted. You may also unlock a single AMPL/M statement, in the same way as described above for block lock.

E. Block Previous

This command is entered by typing <BP>. Since each block command deletes the marks when the block function is completed, there must be a way to restore the block marks for repeated operations on the same block. The block previous command provides that service. If no

previous block exists, or the previous marks have no meaning (after a block delete, for example), no operation will be performed by this command.

F. Block Syncrolink

One of the most exciting uses of the block marks is with syncrolink. The normal operation of syncrolink is to operate from a starting cue until the operator stops the clock by pressing <ESC>. However, when a block is defined, the SUPERSTAR assumes that the user wishes to do an "on-the-fly punch-in/punch-out" operation. This is similar to the operations performed in a sound studio: play back a track up to the punch-in point. Then switch to record mode "on-the-fly". When the punch-out point is reached, return to playback mode, again, "on-the-fly".

This is exactly what the block marks do with syncrolink. This means you can isolate small sections of your program which need to be "re-recorded", in the sense of the wait time and repeat values. The block marks define which program statements are to be syncrolinked. The program can be started at any earlier cue, and progress automatically up to the punch-in point. Inside the block, the user presses the <CUE> key to retime the unlocked waits (or repeats). When the punch-out point is reached, the computer resumes normal operation, until instructed to stop with an <ESC>.

The block syncrolink function also allows the user to syncrolink one of the alternate time lines, by placing the block marks inside a task. This is discussed in section XX.

For more information on the syncrolink function, see section VI and XX.

SECTION IX

INSERT MODE: ENTERING AMPL/M STATEMENTS

A. Introduction

The insert mode of the editor is invoked by typing <I> from the command mode. The COMMAND? prompt and blinking cursor in the mode window are replaced with the prompt "INSERT (REL)", a blank line is inserted at the current line pointer (>) in the program window, and the blinking cursor appears at the first position in the new line.

Mode window in insert mode:

```
!-----------------------------------------!
!AMPL/M EDITOR      11:15 AM        4/15/81!
!                                         !
!INSERT (REL)                  <?> FOR MENU!
!=========================================!
```

Once in the insert mode, the user has access to the insert/edit menu simply by pressing the HELP key. This menu is not available except at the beginning of a program line. The insert/edit menu is shown below:

```
==========================================
!                                        !
!AMPL/M INSERT/EDIT MENU ----------------!
!                                        !
!                                        !
!   EDIT FUNCTIONS:                      !
!                                        !
!      CTRL-A  ACCESS MODE TOGGLE        !
!      CTRL-I  INSERT LINE               !
!      CTRL-S  SHIFT (_,\,[,])           !
!      CTRL-X  CLEAR LINE/DELETE LINE    !
!                                        !
!                                        !
!   CURSOR MOVEMENTS:                    !
!                                        !
!      CTRL-F  FORWARD PAGE              !
!      CTRL-N  NEXT LINE                 !
!      CTRL-P  PREVIOUS LINE             !
!      CTRL-R  REVERSE PAGE              !
!      <-      DESTRUCTIVE BACKSPACE     !
!                                        !
!                                        !
!  <ESC> RETURNS TO EDITOR COMMAND MODE  !
!                                        !
!    <?> RETURNS TO INSERT/EDIT MODE     !
==========================================
```

This menu shows the insert mode CONTROL COMMANDS, which are entered by pressing the <CTRL> key, and, without letting it up, pressing some other key, such as <F>. The <CRTL> key works like a typewriter shift key: it doesn't enter anything itself, but modifies

any other key when it is pressed.

The control commands allow the user to move around in the program to make corrections, additions, and deletions, without returning to the editor command mode.

When in the insert mode, the operator can enter new lines, according to rules described later in this section. When a line is terminated (with a <RETURN> or <SPACE>), the computer will insert the line into the program in memory, execute the statement if the run mode is active, and put a new blank line on the screen automatically. This makes it easy to enter a block of new lines.

Return to the editor command mode is accomplished by pressing <ESC>. The computer will beep, and delete one blank line at the current line pointer. If the current line pointer has been moved to a non-blank line, no line delete will take place.

B. AMPL/M Statements Menu

The AMPL/M statements menu is available from the editor command mode by typing <A>, or from the insert or edit modes by typing <ESC><A>. This menu contains all the possible statements that can be used in an AMPL/M program. The menu is displayed below:

```
==========================================
!                                        !
!AMPL/M STATEMENTS MENU -----------------!
!                                        !
!AA   ABSOLUTE-ACCESS  SH   S/HOME       !
!AD + ADVANCE          SP   SPEED:       !
!AU   AUXILIARY        ST   START TASK:  !
!CU + CUT              TB + TAB:         !
!ER + END: REPEAT      TK   TASK:        !
!ET   END: TASK        UC   UNTIL C      !
!EX   EXTERNAL         UL   UNTIL LAST   !
!FR + FREEZE           UQ + UNTIL CUE    !
!HO + HOLD             WA + WAIT         !
!LC   LET C            WT   WAIT UNTIL T !
!LX + LET X            WX   WAIT X       !
!NA   NO-ADVANCE       1S + 1-SEC        !
!ND   NO-DELAY         12   12-SEC       !
!NO + NOTE:            16   16-SEC       !
!RC   REPEAT C         2S + 2-SEC        !
!RM   REMOTE:          24   24-SEC       !
!RP + REPEAT           4S + 4-SEC        !
!RV   REVERSE          6S + 6-SEC        !
!SC + SOFT-CUT         8S + 8-SEC        !
!                                        !
!  <ESC> RETURNS TO EDITOR COMMAND MODE  !
==========================================
```

All statement names, such as SOFT-CUT, can be entered into the computer by typing the two characters shown in the menu above. The plus signs (+) indicate statements which allow the second character to be a <SPACE>. For example, SOFT-CUT can be entered by typing <SC> or

<S><SPACE>. This allows the non-typist to enter most of the common statements with only a single character (it's easy to find the space bar!).

To exit the AMPL/M statements menu, type <ESC>. This brings you back to the editor command mode. Press <I> to reenter insert, or <E> to reenter the edit mode (the edit mode is described in section X).

When entering a two character statement code, you may press the wrong key. If the key you pressed is a legal entry, the computer will accept it. Otherwise, the illegal entry beep will sound (oh, how you will learn to LOVE that beep!). If you pressed the wrong key(s), and the computer accepted it, just press the backarrow to correct the error. Notice that the second character entered will cause the computer to type out the entire statement name. Likewise, the backarrow will "untype" the same thing. Thus, the computer assisted programming concept comes to the aid of the user again!

After typing the two character statement code, the computer waits for another character input. If you type a backarrow, it will edit the statement name, as described above. If ANY OTHER KEY is pressed, the computer will exit the statement name field, and enter the next field for that statement. These fields will be described later in this section. Once in the next field, it is no longer possible to go back to the statement name field to make corrections! Now, you must use the <CTRL-X> command to clear the line, so you can start over.

Below is a very brief description of each AMPL/M statement. More information is given in appendix I and section XIX of this manual.

1. ABSOLUTE-ACCESS. Toggles the access mode of the dissolve between the normal relative access and absolute access.

2. ADVANCE. Advances the selected projectors.

3. AUXILIARY. Controls the 5 built-in auxiliaries on the Star-3 Programmer or Star Universal Interface.

4. CUT. A 1/20th second dissolve.

5. END: REPEAT. Used to terminate a repeat loop.

6. END: TASK. Used to terminate a task. Section XX covers tasks in detail.

7. EXTERNAL. Control the Star External Auxiliary unit.

8. FREEZE. Locks projectors out of the normal A-B-C sequence.

9. HOLD. Stops the normal dissolve fade at the current lamp intensity.

10. LET C. Used to define the value of a C type variable.

11. LET X. Used to define the value of an X type variable.

12. NO-ADVANCE. Toggles between normal advance mode and the no-advance mode.

13. NO-DELAY. Toggles between the normal delay mode and the immediate advance mode.

14. NOTE:. Used to put comments in the program.

15. REPEAT C. Starts a repeat loop, using a syncrolinked variable (REPEAT C), or a program variable (REPEAT CN).

16. REMOTE:. Turns the remote cue input on or off.

17. REPEAT. Starts a repeat loop, for an indefinite period (REPEAT), or a specified number of loops (REPEAT N).

18. REVERSE. Reverses the selected projectors.

19. SOFT-CUT. A 1/4-second dissolve. Great for fast, smooth animation!

20. S/HOME. Puts the dissolves in standby. The second S/HOME statement in a row homes the projectors. Stands for "standby-home".

21. SPEED:. Sets the maximum speed of cues.

22. START TASK:. Instructs the SUPERSTAR to open a new time line, and to start the specified task running on that time line.

23. TAB:. Used to mark a sequence in the program by name.

24. TASK:. Begins and names a task block. All tasks must be located at the end of the program.

25. UNTIL C. Allows a loop to be terminated based on the results of a value test on a C type variable.

26. UNTIL LAST. Allows exit from a loop at some point besides the last statement in the loop. Can be used to emulate the "end link substitution" capability of the Star Memory.

27. UNTIL CUE. Allows exit from a loop when a <CUE> is received.

28. WAIT. Specifies a preset time delay between cues.

29. WAIT UNTIL T. Specifies a delay until the specified clock time is reached. May be preset and/or syncrolinked.

30. WAIT X. Specifies a delay between cues. May be preset and/or syncrolinked.

31. N-SEC. Triggers an N second dissolve (1, 2, 4, 6, 8, 12, 16, and 24 seconds).

C. AMPL/M Statement Fields

As described briefly above, each AMPL/M statement consists of several FIELDS in a specific order. Each field has its own input and edit rules, and can be entered only from the left. The backarrow can NEVER move from one field to another! Obviously, you CAN move left to right through the required or optional fields in a statement. Also, you can use the backarrow to edit WITHIN a field.

The most common type of statement in a multi-image program is the DISSOLVE statement. A "CUT 1,2,3" is a dissolve statement. All dissolve statements have the same field structure:

<statement-name> {<projector-access>} <screen-area>

The items in the angle brackets are field names. The curly brackets indicate an optional field. Each statement in the AMPL/M language is described in appendix I, using this notation. However, let's cover the most common fields quickly now.

1. <statement-name>: This field was described in detail earlier in this section. Requires a two character input, and allows backarrow edits. Exit from the field with any other key (usually the first key entry for the next field).

2. <projector-access>: This field is ALWAYS optional, and is used to enter projector access. No access field means automatic access will be used. The field is skipped if a number character for the screen-area field is received. Depending on the editor access mode (displayed in the mode window), this field is looking for either the letters A, B, C, representing projectors A, B, C, or the letters C, N, T, representing the CURRENT, NEXT, and THIRD projectors. The access letters must be entered in order, i.e., <CT> is a legal entry (CUR,NXT), but <TC> is not. Backarrow editing is allowed. Exit from the field with a number character (used in the following screen-area field).

3. <screen-area>: This field enters the screen area or dissolve access information into the program statement. Legal entries consist of the digits 1-9 and 0, in that order. Again, entry MUST be in ascending order. Backarrow edits are allowed. Exit from the field with a <SPACE> or <RETURN>.

   If the command is very long, such as a "SOFT-CUT CUR,NXT,THD 1,2,3,4,5,6,7,8,9,0", The editor will compress the screen area numbers by eliminating the commas, to make the command fit on one line of the program. In AMPL/M, no multiple line statements are allowed. The backarrow edit can still be used after a screen-area compression.

All the other fields are covered in detail in appendix I of this manual. Why not try using the definitions above before reading any further? Its best to familiarize yourself with each section as you go, rather than waiting to the end, and trying to remember everything at once!

D. Access Mode

The insert access mode has been briefly mentioned earlier in this manual. The current access mode of the editor is always displayed in the mode window, in parenthesis, immediately after the mode (INSERT or EDIT). The <CTRL-A> key toggles the mode back and forth between (REL) and (ABS). The editor always begins in the (REL) mode.

Mode window in insert mode, relative access:

```
!-----------------------------------------!
!AMPL/M EDITOR     11:15 AM       4/15/81!
!                                         !
!INSERT (REL)                 <?> FOR MENU!
!=========================================!
```

Mode window in insert mode, absolute access:

```
!-----------------------------------------!
!AMPL/M EDITOR     11:15 AM       4/15/81!
!                                         !
!INSERT (ABS)                 <?> FOR MENU!
!=========================================!
```

As described above, the projector-access field accepts different input, depending on the editor access mode. But what, you might ask, does this have to do with the dissolve units access mode? The answer to this may seem a little confusing, at first. Bear with us!

The Star-3 Dissolve access mode is controlled by the EXECUTION of the absolute-access statement, or the standby-home statement. The actual access mode of the dissolve is shown in the status window at all times.

The editor, on the other hand, doesn't care what the mode of the dissolve unit is when you enter the program line. In fact, since the projector-access field precedes the screen-area field, the editor doesn't even know which dissolve the command is for until the statement has been completed!

The purpose of the insert access mode is to allow the user to express in a readable format what he INTENDS his program to do. Thus, if he intends to bring up the "next" projector on screen 4, he enters "CUT NXT 4". By "intends", we mean that he intends that the dissolve will be in that mode when this statement is executed. Now, if he then goes and puts dissolve 4 into absolute-access mode before executing the cut statement, the SUPERSTAR has an interesting problem! The program is doing something different than the expressed intention of the producer!

The only safe time to determine if a possible access error has been made by the user is during run. There are many ways to cause the access mode of the dissolve to be different from that specified in a particular statement, and the program to figure all this out without running would consist of running the program secretly without your knowledge inside the computer! We decided this was silly, so the SUPERSTAR only checks for what it calls ACCESS MODE MISMATCH during a

program run. Thus, in the example given above, the computer would display the message "WARNING: ACCESS MODE MISMATCH" on the center line of the mode window when the "CUT NXT 4" statement was executed. This line is called the ERROR/WARNING line. The warning message will appear for a few seconds, and then disappear.

If you are an experienced Star-3 user, you are probably wondering why the computer should bother checking this out anyway. After all, you might want one program statement to access dissolves in BOTH access modes. To cover this possibility, the SUPERSTAR only issues a WARNING message when an access mismatch is discovered, rather than an ERROR. An error during run always stops the run, while a warning only displays the warning message for a few seconds in the mode window. The access mode mismatch warning should be a great help to new Star System users.

E. Special Character Shift

The editor supports several characters that are not on the Apple keyboard, but are available in the Apple character generator chip. These special characters are useful in notes and names, and are accessed in the insert or edit modes by typing <CTRL-S>, followed by one of the designated keys. A table of the special characters and the designated shift key follows:

type	to get
-	_ (underscore)
/	\ (back-slash)
, or <	[(left square bracket)
. or >	] (right square bracket)

The underscore character is very useful as a "space" character within tab and task names, since a regular space is not allowed. Several examples of names using the underscore follow:

FIRST_SUNRISE

CITY_SQUARE

SAD_PEOPLE

RAIN_CLOUDS

F. Cursor Movements

The AMPL/M editor supports five cursor movement commands in the insert mode. These were already described in section IV. A review follows:

1. <CTRL-N>: Moves to the next line; turns off run.

2. <CTRL-P>: Moves to the previous line; turns off run.

3. <CTRL-F>: Moves one page (15 lines) forward; turns off run. Speed of scroll can be adjusted during the scroll with

the 0 to 9 keys.

4. <CTRL-R>: Moves one page (15 lines) backward; turns off run. Speed of scroll can be adjusted during the scroll with the 0 to 9 keys.

5. <CUE>: Moves one line forward, and issues a cue to the system clock if the run mode is active.

If you move from a blank line with one of the move commands, the blank line will remain in the program. Blank lines may be inserted or deleted at any point in the program, and will have no effect on program execution.

G. Line Inserts And Deletes

The AMPL/M editor allows the user to insert or delete single lines while in the insert or edit modes. Type <CTRL-I> to insert a blank line at the current line pointer, or type <CTRL-X> to delete the current line. Blank lines may be used anywhere in the program to improve program readability, and is especially useful in block replications of program templates.

You may only enter a new line when the current line pointer is pointing at a blank line. Therefore, the blinking cursor will only appear in insert or edit modes when the current line is blank, to indicate line input is allowed.

Note that the <CTRL-X> key has two distinct functions: one is to clear a line, while the other is to delete the line entirely. The AMPL/M editor distinguishes between these two functions by the cursor position: If you are in the middle of entering a new line, then a <CTRL-X> is interpreted as a "clear line" command. Otherwise, the <CTRL-X> is interpreted as a "delete line" command.

H. Program Templates

There are many times when a series of program statements is required, in which several statements are repeated, and often just one statement is different. A simple example is shown below (a ten-screen wipe cut):

```
CUT 1
WAIT 0.14
CUT 2
WAIT 0.14
CUT 3
WAIT 0.14
CUT 4
WAIT 0.14
CUT 5
WAIT 0.14
CUT 6
WAIT 0.14
CUT 7
WAIT 0.14
```

```
CUT 8
WAIT 0.14
CUT 9
WAIT 0.14
CUT 0
WAIT 0.14
```

In other programming systems, you must type in each statement, laboriously. The above sequence would take 100 keystrokes! With program templates, it is much easier! For this example, enter one blank line, followed by a WAIT 0.14 statement. Quit the insert mode, mark the two statement block, and replicate it with a <9BR> command. Go back into insert mode, and enter only the CUT commands, followed by TWO <RETURN>'s each. You have just eliminated about 50 keystrokes!

The more statements that are in the template which are common to each copy, the more keystrokes you save. And keystrokes equals time, and time equals money! This is real COMPUTER ASSISTED PROGRAMMING!

SECTION X

EDIT MODE: MAKING CORRECTIONS

There are two major functions needed in an editor. First, entry of new program lines, and second, making changes to the program once it has been entered. The insert mode provides the first function, while the edit mode and block commands provide the second function.

Actually, the edit mode is almost exactly the same as the insert mode. The only difference is that the automatic blank line insert, which the insert mode makes when invoked and after each new line is entered, is not active in the edit mode. Thus, the user can move around in his program, inserting, changing, deleting lines where needed, without worrying about the editor putting in an extra blank line whenever he enters a new line.

All the same control commands are allowed, including page forward/reverse, line forward/reverse, line insert/delete, line clear, special character shift, and access mode toggle. Also, the insert/edit menu is available at the beginning of any line, simply by pressing the <HELP> key. As you move around in your program, the blinking cursor will only appear at the beginning of blank lines. This is because you can only enter new statements when the current line cursor is at a blank line.

To add a line, move the current line pointer to the desired line, and press <CTRL-I> to insert a blank line. Then type in the desired line. Press <SPACE> or <RETURN> to enter the line into the computer memory, and to execute it if the run mode is active.

To change a line, simply press <CTRL-I> at the incorrect line, type in the new line, and enter it with a <SPACE> or <RETURN>. Then simply type <CTRL-X> to delete the old line.

Other changes in your program can be made using the block edit functions and file read/write functions described in section VIII and XII of this manual.

SECTION XI

PROGRAM DOCUMENTATION

The AMPL/M language provides a means to properly document your multi-image programs. The purpose of program doucmentation is to make it easy to understand what is happening in the program, to make it possible for you to later make changes easily, and to help others to understand the program, if that is desired.

A good example of this is if someone other than the programmer is going to be running the show. The program can contain directions on how to run the show, how to set up the projectors, when to change trays, or when to operate some special equipment manually.

AMPL/M provides three types of statements to allow proper program documentation. They are: Blank lines, Notes, and Tab and Task Names. A sample program segment using some of these features is shown below:

```
204 CUT 1,3,5
205
206 TAB: SECTION_II
207
208 NOTE: ***************************
209 NOTE: SECTION II:  "MAKING IT"
210 NOTE:
211 NOTE: START TAPE RECORDER NOW!!
212 NOTE: ***************************
213
214 WAIT UNTIL T = 0:12:22.00
215
215 NOTE: ANIMATED CHILD PLAYING
216
217 NO-ADVANCE 1,2,3,4,5,6,7,8,9,0
218 WAIT X = 0.15
219 REPEAT C = 12
220  CUT 1
221  WAIT 0.10
```

This example shows one method of program documentation. Keep in mind that the notes take up considerable memory space: for every four characters in a note (or name), one additional cue is used up. You might want to get a 16K ram card, to allow the extra memory space for documentation of large shows. It is very inexpensive memory expansion!

SECTION XII

DISK OPERATIONS

The AMPL/M programming system provides extensive disk input/output facilities. This makes it possible for the multi-image producer to get the most out of his computer. Some of the disk functions are available within the AMPL/M editor, such as program save and load, while others can only be accessed through the main menu. These are called the Disk Utilities.

The disk utilities are available from the main menu by typing <D>. The main menu is replaced with the disk utilities menu, shown below:

```
==========================================
!                                        !
!AMPL/M DISK UTILITIES -------------------!
!                                        !
!                                        !
!   SELECT FUNCTION FROM MENU:#          !
!                                        !
!                                        !
!     B - BACKUP (COPY) DISK             !
!                                        !
!     D - DELETE FILE                    !
!                                        !
!     C - CATALOG DISK                   !
!                                        !
!     R - RENAME FILE                    !
!                                        !
!     L - LOCK FILE                      !
!                                        !
!     U - UNLOCK FILE                    !
!                                        !
!                                        !
!                                        !
!                                        !
!                                        !
!       <ESC> RETURNS TO MAIN MENU       !
==========================================
```

Each of the functions shown on the menu can be invoked with a single keystroke. You can return to the main menu simply by pressing <ESC>. Also, if you select the wrong disk utility, or change your mind before the operation is performed, you can press <ESC> to return from the selected utility to the disk utilities menu.

Most of the utilities work with FILES on the disk. A file is a named block of data stored on the disk. While the file can contain any information at all, with AMPL/M, files only contain multi-image programs. File names may be up to 22 characters long, and must begin with an alphabetic character. Any printable character may be used in the name, with the following exceptions: comma (,), equal sign (=), question mark (?), and slash (/). These characters all have special functions, some of which are for future releases of the AMPL software. The comma and slash functions are described later in this

section. Some example file names follow:

```
                    22 characters
                <---------------------->
NAVA DEMO 1982
MAKING IT HAPPEN VER 2
ANIMATE-FADE
WIPE-CUT LEFT-TO-RIGHT
WIPE-CUT RIGHT-TO-LEFT
```

A complete description of each of the disk utilities provided with the AMPL/M system is given later in this section.

A. Filenames and Automatic File Backup

The AMPL/M editor provides an advanced function called "automatic file backup". This function is handled in the program SAVE utility, accessed when quitting the editor. If you are working on a program with an existing filename (it was loaded from the disk drive, or was saved once before to a specific filename), when you ask to save the program to disk, the SUPERSTAR will offer you the option of using the same name as a default. If you accept the default (simply by typing <RETURN>), the computer does not store the newer version of the program on top of the earlier version, as you might expect! Instead, the earlier versions of the file are renamed automatically, and the new version is given the current default name.

The AMPL/M editor supports up to two generations of automatic file backup, with the use of the filename "extension". This "extension" is appended to the filename when it becomes a backup file. The extension is /A for the most recent backup and /B for the second generation backup. This gives you access to the last two edits of a project automatically, and protects you from destroying a program by mistake - you can always recover from the last save by reloading the /A file!

To make this process completely clear, lets go over it carefully, step by step: The first time you save a file to disk, it will use the provided name, and offer no default. When you re-edit the file, the computer knows that a file exists on the disk with that name, because you just stored it on disk, or you began your edit by loading a file from the disk. When you go to save the new edit, you will now be given the option to save to the same filename by pressing <RETURN> rather than entering a new name. In this case, the SUPERSTAR will go out and rename the existing file on the disk by appending a /A on its name, and then storing the current edit under the filename without an extension. The next save operation will, if the default is taken, go out and rename the /A file to /B, put a /A extension on the last edit, and then save the current edit without an extension. If a /B file exists, it will be deleted automatically, so only two generations of file backups are supported. You may rename the /B file if you wish, before saving the current edit.

This feature will save hours of grief! Several examples of a file with two generations of automatic backup is shown below:

```
          22 characters
      <---------------------->
      APPLE II COMPUTER SHOW
      APPLE II COMPUTER SHOW/A
      APPLE II COMPUTER SHOW/B
      TEST
      TEST/A
      TEST/B
```

Note that the files will not necessarily be listed in the order shown above when a catalog listing is made. The AMPL/M Disk Operating System orders the names of files in the order they are entered into the directory.

Another important point is that, while you can load a backup file with a filename extension, you cannot save it back to the same name! You must give a new name to the edited backup file. This protects you from inadvertently destroying the two generation backup system! Once you have assigned a new name, any additional edits to that new named file will automatically generate the /A and /B files.

The editor also supports other renaming methods you might like to use. When the default name is offered, you may enter the same name with a new postfix, such as version number, or a similar name, by using the forward arrow to accept characters from the default name. This will defeat the automatic backup, and as soon as you hit any character other than the arrow keys, the remaining characters in the name will be eliminated. For example, if you want to number your own backups with version numbers, start with "NAME.V0". On the next edit save, use the forward arrow to type over to the "0", and type a <1> instead. You can thus easily enter your own backup filenames. And each time you save the edit, you are shown the default name with the previous version number!

B. Backup Disk

This function, described briefly in section II of this manual, provides a means to copy any AMPL/M diskette. Because this function requires the use of large blocks of computer memory, the copy utility is "disk resident". This means that the program is not actually part of the AMPL/M system when you boot the system, but is loaded into the computer when needed from the system disk. This means that you must have a copy of the AMPL/M System Master diskette in drive #1 to start a backup.

When you select the backup option by typing <B> from the disk utilities menu, the SUPERSTAR will ask you if you wish to save the current program in memory, if it hasn't been saved already. Type <Y> to save, <N> to not save.

At this point, the disk copy utility will be loaded from the master disk. If the disk in drive #1 is not a copy of the AMPL/M System Master diskette, an error message will be given, and control will be returned to the disk utilities menu.

Once loaded, the copy program first asks you what slot and drive you want to use for both the original and duplicate diskette. For

those of you who are not familiar with the slot/drive designation, the slot number refers to which motherboard connector inside the Apple the disk interface card is plugged into, and the drive number refers to one of two connectors on the interface card itself. If you have a one or two drive system, the slot number will always be the same (it should be 6). Use drive #1 as your "original" diskette drive, and drive #2 as your "duplicate" diskette drive. If you establish this convention at the beginning, you won't be copying old disks onto your latest disk - a real disaster, for sure!

The copy program offers you default values of the slot and drive numbers. To accept them, simply press <RETURN> at each prompt. AMPL/M assumes you have two disk drives, so if you have only one, then enter drive #1 as the duplicate disk as well.

Once you have entered the slot and drive specifications, the SUPERSTAR will ask "ENTIRE DISK OR DATA ONLY (E/D)?". If you press <E>, the entire disk will be copied, including the AMPL/M system itself. In this case, the duplicate disk must be labeled with the Clear Light copyright notice, as described in section II of this manual. If you press <D>, only the AMPL/M program files you have created will be copied. This will generate a data diskette, with no AMPL/M system, and no copyright notice requirements.

You may wish to make up blank data diskettes to use as work masters, rather than copying the entire system onto the work diskette. This eliminates the copyright problem, but requires a separate system diskette and data diskette. This is done by backing up an AMPL/M disk with no program files, or deleting all files after the backup process. Note that you cannot use a standard DOS 3.3 diskette as a data diskette, unless you INIT it to volume number 101, the standard AMPL/M volume number. If you use a separate data disk, put the data diskette in drive #2, and keep a AMPL/M System Master disk in drive #1. There are several "overlay" operations in AMPL/M which require a copy of the System Master in drive #1.

Once you have selected the type of copy you want, the SUPERSTAR will ask you to insert the disk(s) into the selected drives. At this point, it is a good idea to open the disk drive doors, and peek at the disk labels, just to be sure! Close the doors, and press <RETURN> to continue.

If the duplicate diskette already has been formatted, the computer will prompt: "DUPLICATE DISK ALREADY CONTAINS DATA. DO YOU WISH TO CONTINUE (Y/N)?". This is one last chance to make sure you know what you are doing! Answer <Y> to continue.

At this point, the computer will copy the required data from one disk to the other, or, in the case of a single disk copy, will ask the user to swap the needed diskette into drive #1 until the copy process is complete. When the copy is complete, follow the procedures outlined in section II of this manual on disk labeling, or invent your own labeling system. One of the most common mistakes new computer users make (and, sad to say, even some long-time computer users, too!), is to not take the time to carefully label all diskettes, or even to do the backups in the first place! You will get away with this for quite a while, because the Apple is such a reliable machine, but watch out! Just when you can least afford it, Murphy will strike,

and you will be out of a few days work (according to Murphy's law, this should happen about a half-hour before the show for the Board of Directors from IBM!).

During the copy process, the computer may be unable to read or write data to/from a disk drive. This condition will result in an error message which will specify which track and sector the error occurred on. You will be asked if you want to try again. If yes, the copy process will begin at the beginning, including a complete reformat of the duplicate diskette, if a write error occurred. If the same error occurs, compare the track and sector number of the first failure with the second failure. If they are the same, you may not be able to make a copy. Try reseating the diskettes before trying again.

C. Catalog

Each AMPL/M diskette has a special track on it which contains a list of the information on the disk, along with the name and location of each program or FILE. A file is a set of program statements stored on the disk under a user supplied name. Each program you produce with AMPL/M will be stored on the disk as one of these named files.

The catalog function lists the file names for you, so you can see what is on the disk. If you are careful about the names you assign your programs, you will be able to tell exactly what you have on the disk simply by looking at the catalog.

When the catalog function is invoked by typing <C> from the disk utilities menu, the computer will again ask for slot and drive number. This time, only one slot and one drive number is needed. If you want to catalog slot 6 drive 1, simply accept the defaults by pressing <RETURN> twice. If you want to catalog another drive, enter the desired numbers.

The catalog list provides not only the name of your files, but also several other important pieces of information. The catalog list is begun with "NNN SECTORS FREE", which tells you how much more space you have on the disk. In front of each filename, you have a column of SIZE, which tells you how many sectors are used by that file. Also, the file lock status is indicated by an asterisk (*) between the file size and name. Of course, the filenames shown will include any "extensions" added by the AMPL/M file save routine.

An AMPL/M System disk will contain about 300 free sectors when no program files are on the diskette. An AMPL/M data diskette will contain 528 free sectors under the same conditions. Both diskettes will allow up to 105 filenames in the catalog.

On the far right of the screen, after the filename, you will see the date the file was stored on the disk (not the last time the file was read off the disk!). This information will help you keep track of your disk files, especially when you begin to accumulate a large number of files.

If the catalog is too long to fit on the screen, the computer will stop part way through, and leave a blinking cursor at the bottom

left of the screen. If this happens, simply press <RETURN> to see the rest of the catalog. When all of the catalog has been displayed, you will be prompted to press <RETURN> to continue. This will bring you back to the disk utilities menu.

D. Rename File

This utility allows you to change the name of a file. When this utility is invoked, you will be prompted for the old file name and the new file name. Simply type in the desired names. Use the backarrow to correct any mistakes. The <RETURN> key enters the names into the computer. The old name may include a filename extension, but the new name cannot. This allows you to change a backup file into a standard file by renaming it. The filename extension is described earlier in this section.

Make sure never to rename a file to the same name as another already existing file! You will end up with two files with the same name! Actually, you CAN recover from this, by renaming the file again, because the computer will only look until it finds the name you want to change. The renamed file may be the first file rather than the second file you renamed incorrectly, however. It depends on which one came first in the catalog!

E. Lock And Unlock File

The file lock facility allows you to protect a program from accidental erasure. If you try to delete a locked file, the system will give a "FILE LOCKED" error message. Obviously, you can simply unlock the file, and then delete it, but the lock makes you think twice before doing so.

When the lock or unlock utility is invoked, the computer will prompt you for the file name which you wish to lock or unlock. Simply type in the name, using the backarrow to edit any mistakes. Press <RETURN> to enter the name into the computer. No filename extensions are allowed with the lock and unlock command. All backup files are unlocked, but can only be changed by the save utility if the master file associated with the particular backup file is unlocked.

When you lock a file, an asterisk will appear in the catalog listing as an indication of the lock.

F. Delete File

This utility allows you to forever eliminate a file from your disk. You cannot delete a locked file without first unlocking it. Make sure you really want to delete the file, because it is gone for good! Remember, disks are only a few bucks each, so don't delete files unless they are really only junk.

The delete utility works the same way as the lock/unlock utilities: you are asked for a file name, and then the operation is performed. You can use a filename extension with the delete utility, thus allowing you to eliminate backup files.

Since the program files on your AMPL/M disk are DOS 3.3 compatible, you can recover from an accidental file delete if you own a disk utility which has the "undelete" capability. Be careful not to do any other modifications to your disk with such a utility, however, because you might destroy the AMPL/M system track allocations in the VTOC (Volume Table Of Contents).

G. File Read/Write: Building Program Libraries

The file read and file write functions are performed by the AMPL/M editor, from the command mode. These commands are entered by typing <FR> and <FW> respectively.

The file write command is a block oriented command. This means that the user must mark a block to be written to the disk, as described in section VIII of this manual. As with other block commands, the SUPERSTAR will prompt "NNN LINES IN BLOCK OK? (Y/N)". If the number of lines checks with what you thought you marked, type <Y>. The computer will now prompt you for a file name. Of course, you can <ESC> at any point; otherwise, type in the file name, using the backarrow edit function, if needed. Press <RETURN>, and the marked block in your program is safely on disk. Filename extensions are not allowed in a file write.

The file write function does not remove the block from your program. If you wish to do that, simply type <BP> followed by <BD>. This will restore the deleted marks, and then delete the block. For details on these commands, see section VIII of this manual.

The file read function does not require block marks, even though it is also a block command. This is because the computer assumes you want to read in the entire file from disk. The computer also assumes that you have positioned the current line pointer to the location in your program where you want the file inserted. You will be asked for a file name, and when you press <RETURN>, the computer will insert the program segment from disk into your program. Filename extensions are allowed for file reads.

You are probably beginning to realize the amazing power of these two file commands! Yes, they are designed to allow you to build your program from a library of routines! You normally will build your library disk on drive #2, and use drive #1 as your program and system disk. You can swap your library disk in drive #2 whenever needed, if you have more than one library disk. But how can you get the files onto disk drive #2 or from disk drive #2 anyway? Neither of the commands prompts you for slot or drive number! The answer follows.

H. Slot And Drive Numbers In File Names

You may have noticed in the DOS 3.3 manual that file names can have the "S" or "D" options specified after them. This is true with AMPL/M file names, as well. For example, if you want to save a special effect from your program to your library disk in drive #2, mark the desired block, and type <FW>. Answer the OK? prompt, and type in the desired file name. Before you press <RETURN>, type <,D2>, then <RETURN>. This tells the computer to put the file on drive #2!

If you have more than 2 disk drives, you can also use the "S" option. For example, to save the file "ANIMATE-FADE" to slot 5, drive 2, type <ANIMATE-FADE,S5,D2> as the file name. Since the AMPL/M system allows up to 22 character names, plus an optional 2 character extension, the total possible number of characters, including slot and drive specifications, is 30. This is why all filename prompts show 30 "dots" for the filename entry prompt.

I. Default Slot and Drive

The SUPERSTAR Disk Operating System automatically assigns the slot and drive numbers to file names if none are provided by the user. The defaults are the same slot and drive used in the last disk access. The only exception to this rule is when the automatic backup option is taken. When the default filename is shown, the slot and drive on which the file was found is displayed as well. This is the automatic backup default slot and drive number, and isn't necessarily the same as the last slot and drive used. When you accept the automatic backup option, the SUPERSTAR will use the displayed slot and drive option.

SECTION XIII

MULIT-IMAGE UTILITIES

There are a number of useful and required functions which are necessary for any multi-image programming system that are not provided by the AMPL/M editor. Instead, these functions are available from a completely different menu. This simplifies the operation of the editor significantly, and makes the AMPL/M system more "user friendly". These MULTI-IMAGE UTILITIES can be accessed from the main menu by typing <M>. The following menu will be displayed:

```
AMPL/M MULTI-IMAGE UTILITIES ----------

   SELECT FUNCTION FROM MENU:#

     L - LOAD STAR-3 MEMORY FORMAT

     D - DUMP STAR-3 MEMORY FORMAT

     B - BUILD FILE FROM STAR-3 TAPE

     E - EXECUTE CUE GENERATE

     T - TIMING TRACK GENERATE

        <ESC> RETURNS TO MAIN MENU
```

Two of the above utilities are "memory resident", which means that they can be executed directly from the menu without reading information from the AMPL/M system disk. These are <T>: Timing track generate and <E>: execute cue generate. The other utilities are "disk resident", which means that they must be loaded into the computer's memory when called.

If one of the disk resident utilities is requested, the user will be asked if he wishes to save the current program to disk first (assuming he hasn't already done so). After this is done, the user must place a copy of the AMPL/M System Master diskette in disk drive #1, and press <RETURN>, as requested. The selected utility will be loaded, and run. Once completed, the necessary sections of AMPL/M will be automatically reloaded. The entire process only takes a few seconds, so no inconvenience will be noticed.

Each of the utilities will be described in detail below.

A. Loading Star-3 Memory Format

This utility allows you to load a Star-3 Memory dump from tape, via the SYNC-IN jack on the rear panel of the Clear Light Interface card. The dump is translated into an AMPL/M program, and then written to disk. All possible Star Memory sequences are legal. Simply follow the prompts, and play the dump into the computer when requested. The dump can be loaded from tape or directly from the Star Memory via the Star-3 sync-out jack. The latter case is called a program "upload".

When the dump is complete, the SUPERSTAR will ask for a filename, so the dump can be stored to disk. Once this has been done, the AMPL/M system will be reloaded, and the main menu will appear on the screen.

The purpose for this utility is to allow a Star-3 user to upgrade his programs to the SUPERSTAR with the least hassle. All special Star Memory commands, such as "aux-control-repeat" are correctly translated into AMPL/M statements.

B. Dumping Star-3 Memory Format

This utility performs the opposite function as the utility described above. The dump can go to tape, or directly to a Star Memory via the Star-3 Programmer sync-in jack. The latter case is called a program "download". Set up the tape recorder in the same way as described under timing tracks later in this section.

When the DUMP utility is used, special statement sequences must be used in the AMPL/M program to generate those Star Memory functions which don't exist on the SUPERSTAR. These special sequences are generated automatically by the upload utility, and are shown in the table below. The "nnn" refers to the Star Memory timing link.

Star Memory command	AMPL/M statement sequence
AUX-CONTROL-FAST nnn	SPEED: 100/SEC WAIT nnn
AUX-CONTROL-REPEAT nnn	SPEED: 10/SEC REPEAT nnn
AUX-CONTROL-FAST-REPEAT nnn	SPEED: 100/SEC REPEAT nnn
AUX-CONTROL-END nnn	UNTIL LAST WAIT nnn END: REPEAT SPEED: 10/SEC
AUX-CONTROL-FAST-END nnn	UNTIL LAST WAIT nnn END: REPEAT SPEED: 100/SEC

The first manual cue after a "fast" sequence	SPEED: 10/SEC

If you are planning to write an AMPL/M program to download to the Star Memory, use the AMPL/M statement sequences shown above to get the correct translation. Also, make sure you modify all of your wait statements so they have only 3 digits of accuracy - n.nn in fast mode, and nn.n in slow mode. The maximum value is 1.65 or 16.5 seconds. If the translator finds any program statement that it can't correctly translate, an error message will be given. We don't recommend you try to write a program for the Star Memory unless you are familiar with its operation.

When you use this utility, the computer will ask you for an AMPL/M filename, which it will use to load the desired file from the disk. Since this utility is disk resident, the AMPL/M System Master disk, or a copy thereof, must be in drive #1. Set up the tape recorder as described in the section on timing track generation below. When the utility finishes, the AMPL/M system will reload automatically, and control will be transferred to the main menu.

C. Building A File From A Star-3 Tape

This utility allows you to generate an AMPL/M file from a Star-3 program tape, rather than a Star Memory dump. All of the waits will be recomputed, based on the actual time between cues decoded. Thus, while you may have recorded cues on 0.10 increments, the translator will occasionally produce a wait for a value a few hundredths different, because of tape speed variation, or decode time variation. Also remember that loops are not automatically recovered from a program tape! If you generate a hundred cues with a simple loop, this translator will generate a 200 cue program: 100 commands, and 100 waits! Because the memory space in the SUPERSTAR is limited (unlike tape), it is entirely possible that a short AMPL/M or Star Memory program will generate more cues than will fit in the computer's memory. If this happens, you will have to load the program in several passes. Simply follow the prompts, and supply the AMPL/M filename, when requested. The decoded program will be stored under the specified filename.

Since this utility is disk resident, a copy of the AMPL/M System Master disk must be in slot #1. When the utility is done, the AMPL/M system is automatically reloaded, and control is transferred to the main menu.

D. Execute Cue Generator

The execute cue generator is invoked by typing <E> from the multi-image utilities menu. The computer prompts the user to press <CUE> each time an execute cue is desired, or to press <ESC> to return to the menu.

Connect the "SYNC OUT" jack on the computer rear panel to the LINE IN jack of the tape recorder, and put the recorder into record

mode. Press the <CUE> key rapidly, and set the record level to approximately -3 db. Now start the tape, and press the <CUE> key each time you want to record an execute cue on the tape. Usually, this is at the beginning of each major sequence. For details on how to use execute cues, refer to section XVI of this manual.

When you have recorded the desired number of cues, press <ESC> twice to return to the main menu.

E. Timing Track Generation

This utility is used to generate a clock track on tape, and is invoked by typing <T> from the multi-image utilities menu. The user is then asked to enter a starting time for the clock, or to accept the default of "0:00:00.00" (zero hours, minutes, seconds). This option will allow you to begin a clock track at any time you choose, when you are recording sections of a program, such as a mixed canned/live performance, or a module of a show. Starting times can only be set to the nearest second - any value entered to the right of the decimal point will be ignored.

First, you will want to set the record level. Do this by pressing <RETURN> to accept the default starting value. The computer will prompt "PRESS ANY KEY TO BEGIN". Press <RETURN>. This will begin the clock track output. Make sure the tape recorder is plugged into the "SYNC OUT" jack on the rear panel of the computer (the Clear Light Interface Card has 3 jacks, labeled "SYNC IN", "SYNC OUT", and "REMOTE CUE"). Adjust the record level to approximately -3 db on the VU meter (some recorders will work best at a lower level, such as -7 db). Once you have set the volume, press <ESC> to stop. This will return you to the utilities menu. Press <T> to select the clock track function again. Now enter the desired starting time.

When the starting time has been set, the SUPERSTAR will again ask you to press <RETURN> to begin. Start the tape recorder in record mode, and press <RETURN>. The clock track output will begin, and the clock time will be displayed in real time on the computer display. The prompt "PRESS <ESC> TO STOP" tells you what to do when enough clock track has been recorded. Up to 2 hours and 40 minutes may be recorded on a single clock track.

SECTION XIV

PRINTING OUT YOUR PROGRAM

The AMPL/M system provides a means to make a hard copy of your program, using one of the many printers and printer interface cards available for the Apple][computer. The printer interface card MUST be installed in slot #1 on the Apple motherboard.

This option is invoked from the main menu by typing <P>. The SUPERSTAR assumes you wish to print the current program in memory. If no program is in memory, you will be prompted for a file name. Before you press <RETURN>, make sure your printer is set to its top-of-page. The computer will then print out your program for you, just like it appears on the CRT screen, with the addition of page numbers, page headings, and date and time of the print out. The date and time feature will help you in sorting out which print-out is which, a particularly troublesome problem when you get deep into a project!

When the print-out is completed, the SUPERSTAR will automatically return to the main menu.

Most of the common printers on the market will work with the AMPL/M print utility. The requirements are:

1. Some form of speed control on the interface card, either baud rate controlled or handshaking.

2. Control-L form feed character operation.

3. Page length of 66 lines.

4. Page width of at least 40 characters.

5. No "automatic" pagination. If the printer has this option, make sure you turn it off!

Other options, such as auto line-feed, etc., should be controlled by DIP switches either on the printer itself, or on the printer interface card.

SECTION XV

USING CLOCK TRACK

The AMPL/M programming system allows the user to synchronize his program in many different ways. One of the more popular ways is to use a clock track on the program tape. This can be used just in the programming phase of production, where the computer generates a final "show tape" with program cues after the programming phase is completed, or it can be used in both the programming and playback phase of production. If the clock track is used in playback, the SUPERSTAR computer must be present, to convert the clock track signals into a running program. In this case, the actual program cues are stored on a floppy diskette.

Make sure you use the correct speed statement at the beginning of your program, depending on the playback target. Use "SPEED: 100 CUES/SEC" if you are planning to play back the show on the SUPERSTAR. Use the "SPEED: 50 CUES/SEC" if you plan to play back the show with Star Universal Interfaces, and the "SPEED: 10 CUES/SEC" statement if you plan to use Star-3 Dissolves to play back the program.

A. Clock Track And The Internal Chronometer

Section XIII of this manual describes the process of generating a clock track, using one of the multi-image utilities. This is the only step that requires the tape recorder to be in record mode. Once the clock track is laid down, the recorder is simply used in playback mode.

The AMPL/M clock track is simply a series of cues, running at 10 cues per second, which tells the computer what time it is in the show. This information is recorded in hours, minutes, and seconds. Even though the cues themselves are recorded at 10 per second, the computer is able to faithfully "lock" its internal chronometer to within 1/100 second at any point on the tape. This process is called "extrapolation".

The AMPL/M internal chronometer may be running with or without the presence of an incoming clock track. When the computer is first put in run mode, it "autosyncs" the projectors, and prepares to start the internal clock. The operator may start the clock simply by pressing <CUE>, by pressing the remote cue switch (if enabled), or by starting his tape recorder. If the tape recorder is connected to the SYNC IN jack on the rear of the computer, either a clock track signal or an execute cue can start the clock. Execute cues are described in detail in sections XIII and XVI of this manual.

B. Programming With Clock Track

When the computer autosyncs the projectors to a particular cue in the program, the clock is set by adding up relative (X) time values to the most recent absolute (T) clock value in the program. The computed clock value will be shown in the upper right of the mode window.

It is a good idea to start each tabbed sequence with a WAIT UNTIL T statement, to insure unambiguous clock values when autosyncing. If any undefined sequences (WAIT X's with no values, REPEAT C loops with no values, REPEAT UNTIL CUE loops, manually cued sections, etc.) are present in the program since the last absolute time definition, the computer will be unable to determine the correct clock time when autosyncing (Obviously, if you RUN the program from the beginning to this point in the program, it WILL know what time it is). This clock ambiguity can be reduced by the proper placement of WAIT UNTIL T statements. If the computer is unable to determine an absolute time, the clock time will be shown with no values in the hours and minutes position, as ":00.00". More on this strange situation later.

When the computer displays internal clock time in the mode window, it is in normal video. If an external clock track is being received, its time will be displayed in reverse video. This makes it easy to see where the clock value is coming from.

Assuming that the internal clock is NOT running (no manual or external cue has been received since the run mode has been entered), then the computer will do one of two things when a clock track is decoded, depending on the value of the clock track:

1. If the clock track value is LESS THAN the internal clock, the computer will display the message "CLOCK -MM.SS: <CUE> OR WAIT FOR PICKUP" in the mode window. This message tells the operator that he may press <CUE> to force an autosync to the current clock value, or wait until the clock track catches up to the internal clock, at which time the internal clock will start running automatically. This is "on-the-fly pickup". The number of minutes and seconds between the clock values is displayed (MM.SS), but since 59.59 is the maximum value displayed, any greater value will also be displayed as 59.59. The value will count down as the clock track value approaches the internal pickup point.

2. If the clock track value is GREATER THAN the internal clock, the computer will display the message "CLOCK +MM.SS: <CUE> TO AUTOSYNC". This message tells the operator that he MUST press <CUE> if he wants to autosync to the new clock track value. The difference value between the internal chronometer and the clock track will count up as the tape is played.

The purpose for the distinctions made above is to allow the producer to easily work on a section of the program over and over, without having the projectors follow the tape at every instant. First, set the computer to the desired cue, usually with a "goto name" or "tab reverse" command. Autosync the projectors by pressing <R>. Now rewind the tape to the desired spot (you hope). When you start the tape, the projectors will not IMMEDIATELY jump to the new clock value, but rather will wait for an "on-the-fly" pickup. If you don't rewind the tape far enough, the computer will show a clock +MM.SS value, and do nothing. If you rewind it too far, the -MM.SS value will be too large. If you rewind it just right, the -MM.SS value will be small, and will count down to zero. At that instant, the projectors will automatically start functioning at the starting point of this sequence! No need to start earlier, to give the projectors

time to catch up: they will be all ready to go when the pickup occurs.

Once the internal clock is locked to the clock track, the program will proceed with or without the external clock, until stopped by <ESC>. This is because a loss of clock track should not stop the show. If the programmer is present, working on the program, he can simply press <ESC> to stop the clock when he stops the tape.

C. Playback Using Clock Track

Once the programming is completed, the clock track can be used to play back the program. When the show is started, the operator must start the internal clock by pressing the <CUE> key, or set the computer up for an on-the-fly pickup described above. Once running, the show will stay locked to the incoming clock track. If the clock track is "lost", the show will continue on internal clock time. If a clock track time is then received which is different from the internal time, the computer will automatically autosync to the new show time. This allows an operator to stop the tape, move it to some other point in the program, and to restart it, expecting the projectors to automatically resynchronize to the program without any manual intervention at the computer. Don't confuse this operation with CUE SENTRY, the program track automatic resynchronization system used in the Universal Interface: clock track can only be used when the SUPERSTAR computer is present!

D. Making A Program Tape With Clock Track

Once a show is programmed using clock track, it is a simple matter to transfer the program to tape, using the Universal Interface for 50 cues per second on up to 2 tracks, or at 10 cues per second, using the Star-3 Programmer or Universal Interface(s). Simply play back the show, using clock track, but record the SYNC OUT from the Universal Interfaces or Star-3 Programmer, along with the sound track(s), on a second recorder. More information on this operation is given in section XVII of this manual. You will now have a program tape which does not require the SUPERSTAR computer to be played back.

E. Clock Ambiguity

As described above, it is possible to autosync to a cue which has an undefined clock time within the show. This possibility must be taken into consideration when forcing a resync to a clock track. The computer cannot autosync to any undefined segment of program, since it doesn't know what time the cues are to be executed. The computer will "scan forward" in your program, looking for the next WAIT UNTIL T statement, and wait there for the clock track to catch up. This is the reason why there should be a WAIT UNTIL T statement at the beginning of every sequence, when the program will be played back from the clock track. The possibilities get even more interesting with multiple time lines! More on that in section XX of this manual.

SECTION XVI

USING EXECUTE CUES TO SYNCROLINK

Another method of synchronizing a show besides clock track is to use "execute" cues. These cues are generated using one of the multi-image utilities described in section XIII of this manual. This method of synchronizing is used when the playback equipment will be a Star-3 Programmer and Star Memory, and the entire program or several fast sequences will be run directly out of the Star Memory. Once the programming is completed, the program can be "downloaded" to the Star Memory directly using one of the multi-image utilities.

Basically, as described earlier in this manual, the execute cue operates in exactly the same way as a manual cue from the operator. Other systems on the market typically use a 1 KHz. beep on tape to do this. The Star-3 System offers a digital cue instead.

Since the execute cue operates exactly like a manual cue, it can be used to start the internal clock for the purpose of syncrolinking a section of the program. Simply record an execute cue on the tape at the beginning of each sequence. Move the computer to the first cue in the sequence (usually with a "goto name" or "tab reverse" command), and autosync the projectors by pressing <R>. Now the computer is ready to receive a cue from the keyboard, from the remote switch (if enabled by the program), or from tape.

Now type <CS>, to put the clock into syncrolink mode. Start the tape recorder just before the sequence start point on the tape. When the execute cue is received, the internal clock will immediately begin running. Now simply press the <CUE> key for any manual cues or WAIT X cues that need to be timed, as requested in the mode window. Section VI gives the basics for the syncrolink process.

Once the program is syncrolinked properly, it can be played back using the same execute cues. Simply autosync to cue #1, and start the tape running at the beginning. Each sequence should begin automatically when the correct execute cue is decoded. Once the show has been tested in this manner, it is possible to download the program directly into a Star Memory, using one of the multi-image utilities. The program must be limited to the capabilities of the Star Memory, if you plan to do this! This means only 1024 cues, FAST and SLOW sequences, etc.

If any FAST sequences are used, you must use "SPEED" statements to switch back and forth between the slow and fast speed mode of the Star Memory. A list of special statement sequences required to generate the special "aux-control" statements is given in section XIII under the Star Memory Dump utility. Remember, the Star Memory switches back automatically to slow speed mode with any "AUX CONTROL" statement without a "fast" screen, or at a manual cue. The SUPERSTAR only uses the SPEED statements to limit the cue speed generation. The DUMP multi-image utility uses the special "SPEED" sequences to emulate this function of the Star Memory.

The Star Memory has only 0.1 second resolution in slow mode. Also watch the maximum wait values: 16.5 seconds in slow mode, and

1.65 seconds in fast mode. Don't attempt to program for the Star Memory unless you have experience programming it, and have a Star-3 Operators manual handy.

Make sure you leave enough time between the end of one sequence and the beginning of the next, so tape speed variation doesn't kill you! Figure 5% of the length of the sequence preceding the execute cue. For example, after a one minute sequence, allow 3 seconds after the sequence ends before the execute cue for the next sequence. Smaller tolerances may be used if you have a really good tape recorder, or if you plan to use the same recorder to play back the show.

Follow the instructions for a memory download in section XIII, and then try to run the show again, now feeding the execute cue track directly into the Star-3 Programmer. Since the execute cue works the same for the Star-3 System as it does for the SUPERSTAR, the program should work exactly the same.

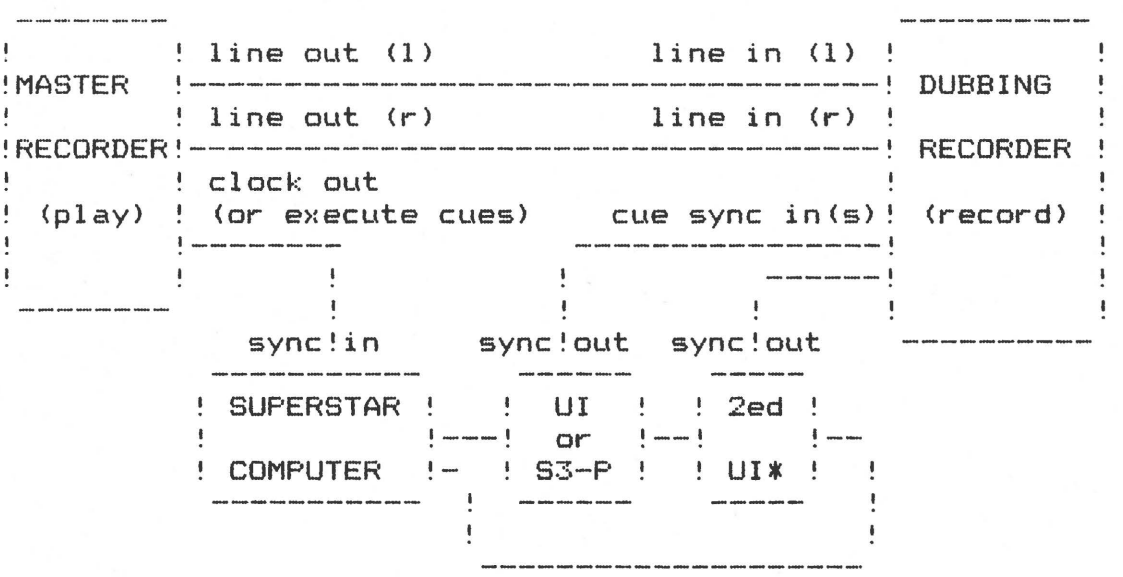
SECTION XVII

STORING YOUR PROGRAM TO TAPE

Once a program has been completed, using either clock track synchronization or execute cues, it often is desirable to put the actual program cues on tape, rather than requiring the SUPERSTAR computer at the playback location. The cues can be recorded using either the Star-3 Programmer or the Universal Interface units. The Star-3 Programmer can only run at 10 cues per second on a single track, so a maximum of 15 projectors can be programmed using it. The Star Universal Interfaces can encode at either 10 cues per second or 50 cues per second, with CUE SENTRY. If two Interfaces are used, 30 projectors can be programmed, on two separate sync tracks.

The program should contain either a "SPEED: 10 CUES/SEC" or a "SPEED: 50 CUES/SEC" statement, depending on which cue encoding speed you choose. Without these statements, the computer cannot generate cues faster than 10 per second, the default speed.

Once the show has been programmed, simply connect the SYNC OUT jack(s) on the rear of either the Star-3 Programmer or Star Universal Interface(s) to the sync track LINE IN jack(s) of the dubbing recorder. Set the record level to -3 db. Also connect the audio track LINE INPUT jacks on the dubbing recorder to the corresponding LINE OUTPUT jacks on the master tape machine. A diagram is provided below:



* Second UI optional with first UI

Set the audio levels, and set both tape recorders to the start of the tape. Goto cue #1 on the SUPERSTAR, enter run mode by typing <R>, and start both tape recorders, the master in playback mode, and the dubbing recorder in record mode. The audio will be copied directly between tape machines, while the clock track or execute cues from the master recorder will go into the SUPERSTAR. The program cues will come out of the Star Universal Interface(s) or Star-3 Programmer SYNC OUT jack(s), and be recorded simultaneously with the audio on the

dubbing recorder.

Once the dubbing has been completed, try running it on the target system: Star-3 Dissolves, Star-3 Programmer and Star Memory, or Universal Interface(s). This is the last check before taking the dub tape into the field.

SECTION XVIII

AUXILIARY CONTROL

The AMPL/M System provides up to 66 auxiliaries for controlling any external device other than slide projectors. Ten of these auxiliaries are "built-in" the Universal Interface units (five each). These auxiliaries are identical to the five built-in the Star-3 Programmer. These auxiliaries are usually referred to simply as AUXILIARY outputs. The remaining 56 auxiliaries are available through the use of the Star External Auxiliary unit, and are usually referred to as EXTERNAL AUXILIARIES.

A. Auxiliary Outputs

The 10 auxiliaries are controlled using the AUXILIARY statement. The statement consists of the statement name field, followed by a screen area field. No access field is permitted. All ten auxiliaries may be accessed simultaneously, or individually. In each bank of five, there are two "momentary" auxiliaries, number 1, 3, 6, and 8. The other three in each bank are "toggle" or "latching" auxiliaries.

The momentary auxiliaries are designed to directly drive a slide projector advance mechanism. They consist of a heavily protected open collector transistor, which can pull to ground up to 1 amp for the 1/2 second duration of the pulse. A maximum of 24 volts AC or DC may be applied. Either AC or DC may be applied to the auxiliary jack, but only the positive portion of the applied signal will be affected by the auxiliary.

The toggle auxiliaries are designed primarily to operate a Clear Light FX Interface unit, which controls 115 VAC devices, up to 10 amps. The auxiliary output is an 11 volt DC signal with a source resistance of 240 ohms. This means that the short circuit current is about 45 milliamps, and the available current at 5 volts output is 25 milliamps. This will drive a 240 ohm load to 5 volts. Any sensitive 6 volt relay can be activated by these auxiliaries.

Since these auxiliaries are latching, each command to them will flip or toggle their state. The first command will turn them on, and the next command will turn them off. A Standby/Home command or autosync will always turn the latching auxiliaries off.

Autosync does not work with the 10 standard auxiliaries. This means that after every autosync, all 10 auxiliary outputs will be off. Keep this in mind when assigning auxiliary channels to external devices.

B. External Auxiliaries

The external auxiliaries are much more flexible than the built-in auxiliaries. There are 7 auxiliary "channels" in a Star External Auxiliary unit. Up to four External Auxiliary units can be daisy-chained together on the same sync track or Star Universal Interface, providing 28 auxiliary channels per bank. The four units

are differentiated by the setting of the "unit select" thumbwheel switch on the External Auxiliary front panel.

The desired auxiliary channel is specified with a three digit number in the AMPL/M "EXTERNAL" statement. This number is called the "BUC" code, which stands for Bank-Unit-Channel. The bank select digit may be 1 or 2, the unit select digit may be from 1 to 4, and the channel select digit may be from 1 to 7. Any other digits placed in the BUC code will be flagged as an illegal BUC code.

Because of the nature of the External Auxiliary commands, only one channel may be accessed in a single AMPL/M statement. The statement specifies not only the BUC code, but also the desired command to the selected channel: OFF, ON, or PULSE. These command modifiers are entered by typing <F>, <N>, or <P>, respectively, and correspond to the "external on" and "external pls/off" switches on the Star-3 Programmer. Several example external statements follow:

AMPL/M Statement	You Type
EXTERNAL (ON) 123	E X N 1 2 3
EXTERNAL (OFF) 247	E X F 2 4 7
EXTERNAL (PULSE) 131	E X P 1 3 1

Each channel may be turned on or off, or, if already off, can be pulsed. Actually, the pulse command is exactly the same as the off command as far as the External Auxiliary is concerned. If either is received when a channel is on, the channel will be turned off. If either is received when the channel is off, the channel will be pulsed for 1/4 second. The PULSE vs. OFF distinction in the AMPL/M statement is allowed for program documentation purposes only.

Unlike the standard auxiliaries, the external auxiliaries will work correctly with autosync. This means that when you autosync to a particular cue in your program, any channel which was turned on earlier in the program and not turned off will be turned on by autosync. The standard auxiliaries will always be off after an autosync.

The External Auxiliary has 7 LEDS on its front panel to display the current status of its 7 channels. Also, there are 8 switches, allowing the user to manually reset all channels, or to toggle individual channels. This greatly simplifies testing of external devices without the computer.

Each external channel supplies normally open and normally closed relay contacts rated for 115 volts and 1 amp. In addition, a FX-compatible DC output is supplied. The different outputs are provided on a 4 pin DIN connector. The normally open relay contact is protected with a MOV (metal-oxide varistor) to suppress arcing of the contacts,when used with slightly inductive loads. The normally closed contacts have no protection, so don't use them with anything but pure resistive loads! For more details, see the External Auxiliary Instruction Manual.

SECTION XIX

PROGRAMMING THE STAR-3 DISSOLVE

The Star-3 Dissolve is a sophisticated microcomputer system, with an extensive command language. This command language is powerful enough to allow the most complex effects to be easily programmed. Because of its incredible operating speed of 100 cues per second, effects not even dreamed of when the Star System was designed can also be easily programmed! The AMPL/M language was designed around the Star-3 Dissolve command set for this reason, rather than requiring a completely new dissolve system, the route taken by all other multi-image manufacturers. Each of the Star-3 commands is described in detail in this section, along with the many automatic features of the dissolve.

A. Dissolve Rates

The Star-3 Dissolve directly executes eleven different dissolve speeds: hard-cue, cut, soft-cut (1/4 second dissolve), 1-sec, 2-sec, 4-sec, 8-sec, 12-sec, 16-sec, and 24-sec. This set of rates was chosen to fit a wide range of desired effects, and can be expanded easily using the HOLD function, described later in this section.

The soft-cut, which was a Clear Light "first" in 1978, allows smooth, rapid animation effects without the annoying flicker associated with cut animation.

When a dissolve command is issued to a dissolve, the dissolve computer automatically determines which projectors should be faded in or out, depending on the current dissolve status. This status is displayed in the CRT display in the status window. The "next" projector will normally be faded up, and all other lit projectors will be faded down. The dissolve will then automatically select a new "next" projector, in the A-B-C sequence. The selected projector is shown in the status window with an "up-caret" beneath the projector letter designation.

This automatic sequencing allows the user to simply enter dissolve statements into the computer, without being concerned with projector access. The dissolve will automatically sequence the projectors for you. For example, to do 4 different dissolves on a dissolve connected to screen area 1, simply enter the following statements:

statements	action
SOFT-CUT 1	brings up projector A
1-SEC 1	dissolves from A to B
4-SEC 1	dissolves from B to C
CUT 1	cuts from C to A

As you can see in the example, you don't have to tell the computer which projectors to fade up and down: it will automatically sequence the projectors for you! Of course you can override the automatic selection of projectors by using ACCESS in the dissolve

statement, described shortly.

B. Current, Next, and Third

One of the unusual features of the Star-3 Dissolve is its logical projector sequencing operation. The sequencer is based on the idea of the NEXT projector, which is defined as the next projector in the A-B-C sequence to be projected onto the screen.

The computer always selects the A projector as the NEXT projector when first turned on, or when placed in the standby mode with the standby/home statement. When the NEXT projector is faded up, usually with the first dissolve statement, the computer immediately scans for a new projector to assign the NEXT status to. This scan is always in the order A-B-C, starting with the current NEXT projector. Only unlit or "dark" projectors are accepted as candidates for NEXT. If no dark projectors are found, then the NEXT status remains unchanged. The AMPL/M system always identifies the NEXT projector assignment in the status window with the up-caret (^) beneath the selected projector.

Once the NEXT projector is known, the remaining projectors are defined relative to it. The projector following the NEXT projector in the A-B-C sequence is called the THIRD projector, while the previous projector in the A-B-C sequence is called the CURRENT projector. AMPL/M abbreviates these designations as CUR, NXT, and THD.

When the first projector begins fading up, it immediately becomes the CURRENT projector, due to the automatic scan for a dark NEXT projector. When the next dissolve statement is received, again, the NEXT projector begins fading up, and immediately becomes the CURRENT projector. In this way, the CURRENT projector designation normally is assigned to the latest projector projecting an image on the screen, while the NEXT projector designation is assigned to the projector containing the next slide in the sequence.

The THIRD projector is normally equivalent to the last or previous projector in the sequence. These concept designations are very important to properly understand, if you want to take advantage of one of the most significant features of the Star-3 Dissolve: program-dependent or relative access.

An example of a sequence of cuts is shown below to illustrate these concepts. Lower case letters are used for normal video projector letters (lamp dark or fading out), while capital letters are used for inverse video (lamp lit or fading up). Assume starting status is from standby mode.

cue	statement	resulting status	status meaning
	(standby)	1> abc ^--	A is dark and NEXT B is dark and THIRD C is dark and CURRENT

```
1   CUT 1      1>       A is lit  and CURRENT
               Abc      B is dark and NEXT
               -^-      C is dark and THIRD

2   CUT 1      1>       A is dark and THIRD
               aBc      B is lit  and CURRENT
               --^      C is dark and NEXT

3   CUT 1      1>       A is dark and NEXT
               abC      B is dark and THIRD
               ^--      C is lit  and CURRENT
```

Notice the sequence of assignments, as well as the A-B-C order of the sequencer. The only case above where the CURRENT projector is not assigned to a projector which is projecting a slide on the screen is when in standby mode, when there are no projectors lit. Study this example carefully, making sure you understand what the dissolve is doing. Otherwise, the discussion following will not make sense!

C. Access Modes

A dissolve statement which does not include any specific access to projectors is referred to as an automatic access statement. This is because it allows the automatic sequencer to select the projectors to be accessed by the statement. If a projector is accessed by a dissolve statement, it will change its status from lit to dark, or from dark to lit. The dissolve rate determines how fast the projector will move to its new status. Thus, a 1-SEC statement will normally access the CURRENT projector to change its status to dark, and the NEXT projector to change its status to lit. While the status of the lamp will change instantaneously when the dissolve statement is executed, the lamp will take one second to attain its new illumination status, assuming it was at full dark or lit condition.

There are many situations where the automatic sequence must be overridden. This is done by specifying which projector should be accessed or affected by the dissolve statement. There are TWO different ways that you can access projectors with the Star-3 Dissolve. One uses the CURRENT, NEXT, THIRD designations described above, and is called "program-dependent access" or "relative access". The other method uses the A, B, C projector designations, and is referred to as "absolute access". Other dissolve systems on the market only allow access similar to absolute access. The Star-3 Dissolve is unique in allowing the more powerful relative access! Each type of access will be described separately.

D. Absolute Access

While the relative access is the normal operating mode of the Star-3 Dissolve, it is more difficult to understand. Therefore, we will discuss the absolute access mode first.

The dissolve must be commanded to enter the absolute access mode before an absolute access-type statement can be executed. This is accomplished by an "ABSOLUTE-ACCESS" statement. This statement can command any of the 10 dissolves in the system to enter (or leave) the

absolute access mode simultaneously. For example, to command dissolves on screen 1, 3, and 5 into the absolute access mode, use the statement "ABSOLUTE-ACCESS 1,3,5". When this statement is executed, the screen numbers 1, 3, and 5 in the status window will turn to inverse video, indicating absolute access mode.

The access mode of the dissolve is toggled on each absolute access statement, so the second time this command is received by a particular dissolve, it will return to the relative access mode. The standby/home statement also returns the dissolve to the relative access mode.

Once the selected dissolves are in the absolute access mode, you must also inform the AMPL/M editor that you wish to enter statements in absolute access mode as well. This is done by typing <CTRL-A> when in the insert or edit modes. The current mode of the editor is shown in the mode window. The prompt "INSERT (REL)" indicates relative access input mode, while "INSERT (ABS)" indicates absolute access input mode.

Each dissolve statement can specify any combination of the projectors to be accessed. The programmer has complete freedom to specify any projectors he wishes, to accomplish the particular effect he is trying to attain. A particular sequence is shown below as an example.

cue	statement	resulting status	meaning
	(standby)	1> abc ^__	
1	1-SEC 1	1> Abc _^_	bring up proj A
2	CUT B 1	1> ABc __^	super B on top of A
3	CUT B,C 1	1> AbC _^_	cut B to C, leaving A on-screen
4	CUT B,C 1	1> ABc __^	cut C to B, leaving A on-screen

As you can see in the example above, you can simply select whichever projector you wish in the statement, for the desired effect. The above effect, a 2-projector "dissolve" with a third projector overlap, can be done much easier using the Freeze statement. However, it serves as a good illustration. Also note how the dissolve computer automatically selects the next available dark projector as the NEXT projector, even if it has to "skip" over an intervening lit projector.

The prime weakness of the absolute access method is related to multiple dissolve usage, and program changes. If you have more than one dissolve in the program, you usually are using access to do 6 or 9 projector effects, or are trying to simultaneously do supers or fade outs on multiple screens. For example, if we are using 15 projectors in a show, assume we have the following status:

```
1>  2>  3>  4>  5>
Abc AbC aBc abC ABc
 ^---^---^-^---^
```

If we now wish to superimpose a 5-screen title over the current projectors simultaneously, we have a problem! No single statement can do it! Other systems offer "presets", which gets around the simultaneity problem, but still takes up to four cue slots of 1/10 or 1/20 second each! Clear Light developed the relative access concept to solve this problem.

Before describing relative access, lets discuss the other major shortcoming of absolute access. Because of the time-consuming preset problem mentioned above, the user is often forced to figure out how to get a particular projector status set up for a special effect by modifying the previous automatic sequence into a accessed sequence. A program with a lot of animation effects is usually done almost completely with access-type statements. Later, if you wish to add or delete a slide or two, you have to practically rewrite the whole program! All the accesses are wrong! Relative access eliminates this problem completely!

E. Relative Access

Relative access uses "concept" addressing of projectors rather than hardware addressing. Absolute access refers to the actual hardware connections between the dissolve unit and the three projectors to which it is connected, via the A, B, and C projector cables. Relative access, on the other hand, refers to the projectors by their status designations CURRENT, NEXT, and THIRD. While it will not be immediately apparent, this approach is extremely powerful, but not as easy to understand. Please make the effort to understand the concepts behind relative access. Otherwise, you won't be able to get the full power out of your computer system!

The relative access concept can be illustrated in the following way: Pretend you have three stickers, labeled CURRENT, NEXT, and THIRD. These stickers are attached to projectors C, A, and B respectively, when the dissolve is in standby mode. When the first dissolve statement is executed, and the A projector is faded up, the computer reassigns the CURRENT, NEXT, THIRD designations. Remove the stickers, and place them on the A, B, and C projectors respectively. As you continue to do dissolves, the stickers will "rotate" around the A-B-C projector loop. The stickers tell you which projector is which, according to the relative access designators. Once you get used to the designation rotation, you will only need to see the NEXT up-caret (^) in the status display to understand the situation.

The reason this access mode is called "relative" or "program dependent" is because the projectors accessed by any given statement

will be determined by the current status. Notice that the absolute access mode does not depend on status at all. You always use A to access the A projector. With relative access, you tell the computer what you want to accomplish on the screen, and it selects the correct projector for you!

Lets look at the 15 projector example given above in the absolute access section:

```
1>  2>  3>  4>  5>
Abc AbC aBc abC ABc
_^___^___^_^_____^
```

With relative access, the desired superimposition can be accomplished with a single statement! Notice that each screen has a dark NEXT projector, designated by the up-carets. The statement "CUT NXT 1,2,3,4,5" will do a 5-screen superimposition, by fading up the 1B, 2B, 3C, 4A, and 5C projectors. In fact, the same statement will work with almost any status! If you took out one slide in screen 1 prior to this sequence, so that projector C was lit and A was next, the same program statement would perform the superimposition! Of course, when you removed the slide, you moved all of the other slides in the trays, so the statement automatically locates the correct slide in its new projector and tray position! Amazing! Saves hours and hours of reprogramming time!

The relative access mode is great for "concept" effects, such as "fade all current projectors to black", or "superimpose the next projector on 3 screens", because it is a "concept" access mode. The correct projector is automatically selected by the computer, even when slides are added or removed in previous sequences. Also, no time-consuming "presets" are required!

When using relative access statements, the AMPL/M editor must be in the relative access mode. The <CTRL-A> key toggles between the (REL) and (ABS) input modes. If you use relative access on a dissolve which is in absolute access mode, or vise versa, the SUPERSTAR will warn you of a "ACCESS MODE MISMATCH". This is a warning, not an error, because it is sometimes desirable to use one statement to access dissolves in both modes. This is very tricky, so don't do it unless you are sure of what you are doing. The dissolve will interpret the access according to its mode, regardless of the mode displayed in the AMPL/M statement. The following table described the interpretation:

access used	dissolve mode ab-accs	rel-accs
A	A	CUR
B	B	NXT
C	C	THD
CUR	A	CUR
NXT	B	NXT
THD	C	THD

F. Tray Movement Statements

When the Star-3 Dissolve is turned on, it enters the "standby" mode, which means that all lamps are out, and all the optional modes of the dissolve are turned off. These modes include the absolute-access mode discussed above, plus the no-advance, no-delay, and 2-projector modes. The standby mode can be restored at any point in the program by issuing a S/HOME commands to the dissolve. If two S/HOME commands are received in a row, the dissolve will also home the slide trays to their starting position, by the shorter route.

In normal operation, the Star-3 Dissolve will automatically advance a slide projector 1/2 second after a fade-out completes. This advance is forced to occur earlier if a command to fade up is received before the fade out or 1/2 second delay is completed. The NO-ADVANCE statement will turn off the automatic advance after every fade out, allowing animation effects, or reversals of fade direction and rate without any projector tray movements. The no-advance mode is toggled on or off on each NO-ADVANCE statement. Also, the no-delay mode is turned off by this statement.

The 1/2 second delay provided by the dissolve allows the lamp filament to cool down enough so that shutter chop is totally invisible on the screen, even on fade-to-black. However, it is often desirable to eliminate this delay, for faster sequencing. The Star-3 Dissolve is the only dissolve on the market which allows this to be done automatically! The 1/2 second delay is turned off by the NO-DELAY statement. The no-delay mode is also toggled on and off by this statement, and the no-advance mode is cleared as well.

The Star-3 Dissolve also allows you to program either advances or reverses by using the ADVANCE or REVERSE statements. These statements can use automatic access, which is interpreted by the dissolve to mean ADVANCE (DARK> or REVERSE (DARK). This cycles the trays on all projectors on the selected dissolve which are dark or fading down, a real convenience!

The user may also use either type of access with the REVERSE and ADVANCE statements, to select whatever projectors he wishes to cycle.

G. Automatic Overlap

Another of the unique features of the Star-3 Dissolve is called Automatic Overlap. This feature allows the dissolve unit to receive dissolve commands before completing the previous dissolve, without causing a mess on the screen. This capability has been partially copied by other manufacturers, but, fortunately for us, they didn't fully understand what the feature does, so they only copied half of it!

Automatic overlap only works when an automatic access dissolve statement is used, i.e., no access specified. In this case, any projector still fading up from the last dissolve command will CONTINUE to fade up, to the half-way point between 100% illumination and whatever percent illumination the lamp was lit when the new dissolve command was received. For example, if you issue a 4-SEC dissolve command, and, 2 seconds later issue another 4-SEC dissolve command,

the projector fading up from the first dissolve command is only at 50% illumination when the second command is received. That lamp continues fading up until it reaches 75% illumination - half way between 50% and 100%. When it automatically reverses direction at the 75% illumination level, the next projector fading up is at its 25% illumination level. This means that the two projectors will meet at the 50% level, exactly what you want for a proper dissolve! Other systems copy the reversal of fade direction, but don't delay the reverse as does the Star-3 Dissolve. This results in a intensity crossover at some level below the ideal 50% level.

The automatic overlap function thus allows you to do "continuous dissolves", where there is always a dissolve in process on the screen. Three projectors are required, minimum. The screen intensity remains constant, and you get a constant flowing effect. For example, you can run 8-SEC dissolves every 3.5 seconds, and get a continuous, flowing dissolve effect! An example program to get this effect is shown in the following example:

```
REPEAT 10
 8-SEC 1
 WAIT 3.50
END: REPEAT
```

A table showing the maximum speed of continuous dissolves with three projectors is given below, assuming normal and no-delay modes. The mathematical formulas for calculating these numbers are fairly complex, so these numbers were derived from experimentation.

	dissolve mode	
dissolve speed	normal	no-delay
cut	0.75	0.50
soft-cut	1.00	0.65
1 second	1.25	0.95
2 second	1.75	1.40
4 second	2.50	2.25
6 second	3.30	3.15
8 second	3.50	3.50
12 second	5.25	5.25
16 second	7.00	7.00
24 second	10.50	10.50

Note that the no-delay mode only improves the faster rates. Also, keep in mind that the maximum rate of continuous dissolve will depend on the type of slides being projected. With certain slides, you will have to go slower that the above rates because pop-on and shutter-chop will be visible on the screen.

While automatic overlap and its resultant continuous dissolve effect is very useful, it is sometimes desirable to "override" this feature. This can be done by using relative access to perform the dissolve command, rather than automatic access. The access "CUR,NXT" will fade out the CURRENT projector, and fade up the NEXT projector in the same way the automatic access command will, except that automatic access will not operate. In this case, any projector fading up will IMMEDIATELY begin fading down when another fade command is received, directed at that projector. Any AMPL/M dissolve command with either

access mode will override the automatic overlap feature.

It should be pointed out what the dissolve will do if it receives a fade up command for a projector which is fading out. Assuming the normal advance mode is active, the fading down projector is set up to advance automatically at the end of its fade. When the fade out is "stepped on" by another command to fade up, the projector is advanced immediately, and the lamp is cut off, to begin its fade up from the 0% illumination level. This process insures that tray synchronization is not affected by the timing of the show cues, but only by the dissolve modes and status.

H. Ripple Dissolves

Above, we briefly discussed what the dissolve does when the projector is fading out in normal advance mode, and a new dissolve command is received. If the advance is turned off by a NO-ADVANCE statement, then an interesting thing happens! Rather than cut to 0% illumination, the fade direction reverses, and the new rate specified by the new dissolve command is used in the new direction. This allows the user to change direction and rate of fade at any point in the fade ramp, and can be used to produce "ripple" dissolves. A sample program loop to produce a ripple dissolve is shown below. Absolute access is used in this example.

```
NOTE: PROJ 1A LIT, 1B DARK
NO-ADVANCE 1
WAIT 0.10
REPEAT 8
 2-SEC A,B 1
 WAIT 0.50
 4-SEC A,B 1
 WAIT 0.50
END: REPEAT
2-SEC A,B 1
```

In the example above, the projector A is faded down for 0.50 seconds at a rate of 2 seconds, which means from 100% to 75% illumination. Then, it is reversed, and faded up, at a new rate of 4 seconds, again for 0.5 seconds. This brings it up from 75% to 87.5% illumination. For each loop, the ending illumination is 12.5% lower, but there is a see-saw effect, called a ripple. After 8 seconds of that, the dissolve is finished. The 2-SEC statement after the loop insures an odd number of dissolve statements to make sure the lamps stay at their new status, rather than returning to the old status at the 4 second rate.

I. The Hold Statement

Sometimes it is desirable to stop a dissolve in process, and hold it there for a while. This effect is generated using the HOLD statement. If a dissolve is in process, an automatic access HOLD statement (no access specified) will stop all projectors that are fading. If the HOLD is repeated, all projectors will be released from the hold, and continue their fade. The hold function can be used at any point in the fade ramp, as many times as desired. An example

follows:

```
NOTE: BEGIN 4-SEC DISSOLVE
4-SEC 1
WAIT 1.00
NOTE: HOLD AT 25% LEVEL
HOLD 1
WAIT 2.00
NOTE: CONTINUE DISSOLVE
HOLD 1
```

If access is specified in the HOLD statement, then only the specified projectors will be affected. Since the hold function is a toggle function, it will alternate between on and off on each HOLD statement, on each projector accessed. Actually, there are two ways to release a hold. The first way, described above, is to execute a second HOLD statement to the desired projector. The second way is to use any NN-SEC statement to the desired projector. In this case, not only will the hold be released, but the direction of the fade will be reversed, and the fade will be at the new rate. An example is given below. Assume that the dissolve is in absolute-access mode, and no-advance mode:

```
NOTE: BEGIN 8-SEC DISSOLVE
8-SEC A 1
WAIT 4.00
NOTE: HOLD IT 2 SEC AT 50%
HOLD A 1
WAIT 2.00
NOTE: REVERSE FADE AT NEW RATE
4-SEC A 1
```

The HOLD statement thus allows you to make any sort of fade ramp you desire. Also, you can lengthen any dissolve rate, as well. For example, to do a 48 second dissolve, use the following sequence:

```
24-SEC 1
WAIT 0.10
REPEAT 480
 HOLD 1
 WAIT 0.10
END: REPEAT
```

The loop count is computed by dividing the total time (48 seconds) by the loop time (0.1 second). The 24 second dissolve is "modulated" by the hold loop, and is doubled. Any other multiple can be used as well. To get 36 seconds, use a 1.5 multiplier on the 24 second dissolve:

```
24-SEC 1
WAIT 0.10
REPEAT 240
 NOTE: TURN HOLD ON 1/20
 HOLD 1
 WAIT 0.05
 NOTE: TURN HOLD OFF 2/20
 HOLD 1
 WAIT 0.10
```

```
END: REPEAT
```

In this case, the on/off ratio is 2 to 1, giving a 50% increase in ramp time. The loop counter is computed by dividing 36 by 0.15, the loop time.

Another useful function of the HOLD statement is to generate the "animate-fade" effect. This is covered in section XX of this manual.

J. The Freeze Statement

Another unique feature of the Star-3 Dissolve is the FREEZE statement. Don't get this statement confused with the HOLD statement described above. Unfortunately, different manufacturers use different nomenclature, and our HOLD is someone else's FREEZE. Our FREEZE is totally unique.

The FREEZE statement has several important uses. Its basic function is to modify the access computations within the dissolve. The implications of this modification are very extensive, so try to understand what is happening with this statement. If you do, you will find yourself writing programs with very few statements which do very powerful sequences!

When a projector is frozen with the FREEZE statement, the automatic sequencer will not select it as the next projector, and thus will always skip over the frozen projector. This means that the remaining two projectors will operate as a 2-projector dissolve! This is great for 2-projector ripple dissolves, without access. For example, to do a ripple dissolve between the CURRENT and NEXT projectors, use the following sequence:

```
NOTE: LOCK OUT THIRD PROJ
FREEZE THD 1
WAIT 0.10
NOTE: BEGIN RIPPLE
REPEAT 10
 2-SEC 1
 WAIT 0.40
 1-SEC 1
 WAIT 0.10
END: REPEAT
NOTE: COMPLETE DISSOLVE
2-SEC 1
WAIT 0.10
NOTE: TURN OFF FREEZE
FREEZE (OFF) 1
```

The FREEZE (OFF) statement is entered as a automatic access freeze, and the AMPL/M editor inserts the OFF for program clarity. Note that the freeze function is NOT a toggle. More on this shortly.

When a projector is placed in the freeze mode, this is displayed for the user in the status window. Normally, there is a dash (-) under each projector letter, or an up-caret (^) if the projector is NEXT. If the projector is frozen, then a plus sign (+) appears under the projector letter. The addition of the vertical line indicates

that the sequencer cannot access that projector.

Another use for projector lockout is for tray changes. The dissolve will skip over the frozen projector automatically, without using lots of access statements. In fact, you can still use multi-screen relative-access functions for simultaneous operations, even when one projector is frozen! For example, to fade-to-black the CURRENT projectors on 3 screens, when screen 2 has one projector in freeze for a tray change, use the statement "NN-SEC CUR 1,2,3". This is the same statement used anywhere else in the show! Other systems require special programming to skip over the projector, with elaborate presets and access calculations on the part of the programmer.

If the FREEZE (OFF) statement is executed when the dissolve is in standby mode, a special function in the dissolve is activated, called 2-projector mode. This is NOT the same as freezing one of the three projectors! The C projector is completely locked out of the sequence, and cannot be accessed under any conditions. In fact, the SUPERSTAR will only display the A and B status when the 2-projector mode is activated. This mode is provided to allow the use of Star-2 programs on Star-3 Dissolves without having to reset the 2/3 projector switch on the dissolve. If you plan to use 2-projector dissolve programs, use the FREEZE (OFF) statement as the first statement in your program, and it will play on both types of Star Dissolves, without manual intervention.

When the 2-projector mode is activated, there is an automatic reassignment of the THIRD access function. The relative access is "collapsed" onto the two remaining projectors. Both CUR and THD access the same projector, and NXT access the other projector. The 2-projector mode is cleared by a S/HOME statement.

As mentioned above, the FREEZE statement is not a toggle function. This is because the access is "collapsed" when one or more projectors are frozen, in the same way as in 2-projector mode. A table is given below to describe this function:

1 PROJ UNFROZEN: CUR = NXT = THD

2 PROJ UNFROZEN: CUR = THD, NXT

3 PROJ UNFROZEN: CUR, NXT, THD

This collapse is important to understand, especially when using multi-screen access statements, and some of the screens have frozen projectors. Also, if you use a FREEZE statement with relative access, you can only freeze a projector which is unfrozen, because the access only goes to unfrozen projectors. When the statement is executed, the selected projectors will be frozen, and all previously frozen projectors will be unfrozen. Look at the following example:

```
NOTE: TRAY CHANGE LAST PROJ
FREEZE THD 1
(more program here)
NOTE: TRAY CHANGE COMPLETED
NOTE: SETUP FOR NEXT TRAY
FREEZE NXT 1
(more program here)
```

```
NOTE: TRAY CHANGE COMPLETED
NOTE: SETUP FOR NEXT TRAY
FREEZE NXT 1
(more program here)
NOTE: ALL TRAYS CHANGED
FREEZE (OFF) 1
```

In the above example, the last (THD) projector prior to the current projector is put into freeze mode for the tray change. In the second and third case, only two projectors are available, so the NXT access is used to select the dark projector.

While relative access is collapsed to avoid frozen projectors, the absolute access is not. The A, B, and C access selections will still access the A, B, and C projectors automatically. The automatic sequencer will still skip over frozen projectors, but the program can still access them with absolute access type statements. If absolute access is used in a FREEZE statement, the selected projectors will end up being frozen, and the other projectors will end up unfrozen, no matter what the previous freeze status was.

A very interesting result of the access modification operation with the FREEZE statement is related to the FREEZE NXT statement. Since a frozen projector cannot be NEXT (because it is locked out of the automatic sequence), the dissolve computer is forced to reassign the NEXT designation to another projector. Thus, the FREEZE NXT statement can "bump" the NEXT designation without actually turning on or off any projector lamps! This provides some amazing programming features. One example is given below, which allows one bank of projectors to run in the reverse sequence (C-B-A) while, with the same statements, another bank is running in the normal A-B-C sequence:

```
NO-ADVANCE 1,2
WAIT 0.10
REPEAT 30
 NOTE: REVERSE SEQUENCE ON 1
 FREEZE NXT 1
 WAIT 0.05
 CUT 1,2
 WAIT 0.05
END: REPEAT
FREEZE (OFF) 1
```

The reason the above sequence works is that the FREEZE NXT statement forces the NEXT designation to move to the THD projector, which is equivalent in concept to the PREVIOUS projector. Since the previous projector actually becomes NEXT, the sequence runs backwards!

The FREEZE statement can also be used to simulate the "preset" commands on other multi-image programmers. Two examples are given below:

1. Current status: 1 CUR and 2 THD lit. To fade to black both without using absolute-access:

```
NOTE: MOVE NXT POINTER TO THD
FREEZE NXT 2
WAIT X
CUT CUR 1,2
WAIT X
NOTE: TURN OFF FREEZE
FREEZE (OFF) 2
```

2. Current status: 1 CUR,THD and 2 CUR lit. To fade to black all three with one command:

```
NOTE: COMPRESS THD TO CUR
FREEZE THD 2
WAIT X
NOTE: CUT ALL THREE PROJ
CUT CUR,THD 1,2
WAIT X
NOTE: TURN OFF FREEZE
FREEZE (OFF) 2
```

The ability to "bump" the NEXT projector designator comes in handy when you edit a sequence, and remove or add several slides. Before you do the edit, go to the first sequence after the one you intend to edit, and autosync the projectors to get the correct status. Now, enter a note in the program giving the current status for NEXT and LIT on all relevant screens. After you edit, autosync to the note again, and check the status. If you need to move NEXT, use the FREEZE statement. This will eliminate swapping slides between trays!

Sometimes it is necessary to shorten an animation sequence slightly. If it is a 3-projector sequence, normally you must shorten it in steps of three. With the FREEZE statement, you can shorten it in steps on one, and force the correct status for the next effect by a FREEZE NXT statement!

An often forgotten capability of the Star-3 dissolve is the ability to control a frozen projector by using absolute-access, mentioned earlier in this section. This allows some really interesting capabilities! You can fade up, fade down, advance, or reverse the frozen projector while doing 2-projector effects, by using any absolute-access statement addressed to the frozen projector. If you remember this capability, you will find uses for it at the most crucial points in complex sequences.

There are many other amazing things you can do with the FREEZE statement. We'll leave the creative thinking up to you! Happy adventuring!

K. Speed Restrictions

The Star System can run at 100 cues per second if the program is run out of memory. Also, the Star-3 Dissolve can run at 100 cues per second for short bursts, when operated on 60 Hz. power. However, when 50 Hz. power is used, you cannot use 100 cues per second on a single dissolve without occasionally losing a command. You can run the system at that rate, as long as any given dissolve does not receive

its commands faster than 50 per second. At 60 Hz., the Star-3 Dissolve can operate at a sustained 100 cues per second for 5 or 6 cues.

SECTION XX

ADVANCED PROGRAMMING TECHNIQUES

The AMPL/M programming language offers the multi-image producer not only the most flexible and powerful programming environment, via the AMPL/M editor and AMPL/DOS, but also the most powerful language constructs for simplifying the translation from ideas to program reality. The three areas where the greatest advance in language concepts has occurred are in repeat loops, program variables, and multiple time lines. Each of these areas will be discussed in this section.

A. Repeat Loops

Many examples of repeat loops were given in the previous section. The AMPL/M language allows a number of interesting and powerful extensions to the standard repeat loop. For example, the loops can be "nested" up to four deep. This means that loops can be placed inside loops. This does not refer to how many loops you can have in your program, or how many times a loop can repeat. Lets look at an example of a nested loop:

```
NOTE: INITIALIZE THE DISSOLVE MODE
NO-ADVANCE 1,2,3
WAIT 0.10
NOTE: REPEAT THE EFFECT 10 TIMES
REPEAT 10
 NOTE: BRING UP A SET OF IMAGES
 1-SEC 1,2,3
 WAIT 1.00
 NOTE: DO 5 ANIMATIONS ON THEM
 REPEAT 5
  NOTE: EACH ANIMATION CONSISTS OF
  NOTE: 3 WIPE CUTS AND A WAIT
  REPEAT 3
   CUT 1
   WAIT 0.10
   CUT 2
   WAIT 0.10
   CUT 2
   WAIT 0.50
  END: REPEAT
  NOTE: FOLLOWED BY A SOFT-CUT
  SOFT-CUT 1,2,3
  WAIT 0.50
 END: REPEAT
 NOTE: GET RID OF SLIDES FOR NEXT
 1-SEC CUR 1,2,3
 WAIT 1.50
 ADVANCE (DARK) 1,2,3
 WAIT 1.00
END: REPEAT
```

In the above example, heavily notated, we use three "levels" of nesting. This is shown graphically by the indent levels in the example. AMPL/M automatically indents each loop level, to make it easy to see the range of a loop.

Other systems offer fixed repeat counts, as well as cued repeat counts, either at programming time, or run time. AMPL/M offers these types of loops as well, with some new twists. For example, to get a repeat loop to run until you cue it at run time, use the following sequence:

```
REPEAT
 UNTIL CUE
 (loop statements)
END: REPEAT
```

This is what we call the "repeat until cue" loop. It actually even says that in your program! But wait! Notice that the UNTIL CUE is actually a separate statement from the REPEAT. This is very important. The REPEAT statement without a count value is an "indefinite" repeat loop, which must be terminated by some condition. In the example above, that condition is a manual cue. Other UNTIL statements will be described shortly. Since the UNTIL statement is separate from the REPEAT statement, it can be placed anywhere in the loop, not just at the beginning. In fact, you can have any number of UNTIL statements inside a loop. Look at the following example:

```
REPEAT
 CUT 1
 WAIT 0.10
 UNTIL CUE
 CUT 2
 WAIT 0.50
END: REPEAT
```

The loop in the above example will run until a manual cue is received, but will exit at the UNTIL statement, not at the END statement. This means that the CUT 1 will be both the first AND the last cut to execute in the loop. The UNTIL statement actually causes the computer to skip over the CUT 2 and WAIT 0.50 statement in the loop when a manual cue is received! This capability for exit from a loop at any point within is extremely powerful.

Another UNTIL statement is UNTIL LAST. This statement causes an exit from the loop when the computer is running through the loop for the last time. Look at the example below:

```
REPEAT 10
 CUT 1
 UNTIL LAST
 WAIT 0.10
END: REPEAT
```

In this example, ten cuts and only 9 waits will be executed in the loop, since the loop will exit at the UNTIL LAST on the tenth repeat. The next statement in the program immediately after the END statement will be executed at the exact same time as the WAIT statement would have executed if no UNTIL had been used. This loop

structure allows simulation of the "end-link substitution" capability of the Star Memory. In the example above, the "end-link" would be placed immediately after the END: REPEAT statement (a WAIT statement, of course). On the last loop, the end-link WAIT will be used rather than the WAIT inside the loop.

Since the UNTIL LAST can also be used at any point in the loop, other interesting effects can be generated as well. For example:

```
NOTE: DO THE EFFECT 10 TIMES
REPEAT 10
 NOTE: DO 5 OVERLAPPED CUTS
 REPEAT 5
  CUT NXT 1
  WAIT 0.15
  UNTIL LAST
  CUT CUR 1
  WAIT 0.25
 END: REPEAT
 NOTE: FADE THE FIFTH AT 1/4
 SOFT-CUT CUR 1
 WAIT 0.50
END: REPEAT
```

In the above example, we did 5 overlapping cuts, but on the fifth, instead of cutting it out, we exit the loop, and fade it out with a SOFT-CUT instead! Since the UNTIL statement takes no program time to execute, the SOFT-CUT takes place at exactly the same instant that the CUT would have without the UNTIL statement.

Lets look at one more example of the use of UNTIL LAST:

```
REPEAT 10
 REPEAT 6
  CUT 1
  UNTIL LAST
  WAIT 0.25
 END: REPEAT
 UNTIL LAST
 WAIT 1.00
END: REPEAT
WAIT X
```

In this example, we do 6 cuts quickly, and then wait one second. We repeat this sequence 10 times. On the last group of 6 cuts, we exit out of the inside and outside loops via the two UNTIL LAST statements. This makes the computer execute the next statement after the loop (WAIT X) without doing either the WAIT 0.25 or the WAIT 1.00. Notice that TWO separate UNTIL LAST statements are needed: one to exit from each nested level.

The AMPL/M language has a third UNTIL statement (UNTIL Cn = 0), which will be covered later in this section.

If you wish to set the length of a loop at programming time, rather than at run time, you can use the syncrolink timing system described earlier in this manual, with a REPEAT C loop. The loop counter is set when you press <CUE>, and can be locked to the set

value, if desired. This allows you to adjust the loop running time to match a sound track, without having to compute the exact time of the loop or the sound track event.

REPEAT C loops can also be nested, and may use UNTIL statements to cause loop termination at any point within the loop. Nested REPEAT C loops can also be syncrolinked, even with WAIT X statements within them. Single level REPEAT C loop syncrolink was covered in section VI of this manual. Lets look at an example of multiple level loops:

```
10  REPEAT C
11   CUT 1
12   WAIT X
13   CUT 2
14   WAIT 0.25
15   REPEAT C
16    CUT 3
17    WAIT X
18   END: REPEAT
19   CUT 1,2
20   WAIT X
21  END: REPEAT
```

In the example above, it might be difficult to figure out how the cue requests will be entered, but because the SUPERSTAR prompts you for each type of cue separately, you should have no problem. The order of cue requests in the example above are shown in the table below. Requests which are invisible because they are superceded by another request instantly are shown in parenthesis:

type	from cue #
(C-type)	10
X-type	12
C-type	10 (for 1/4 sec)
(C-type)	15
X-type	17
C-type	15
(C-type)	10 (if not cued already)
X-type	20
C-type	10 (if not cued already during the 1/4 sec)

As you can see, for deferred cue requests, the "deepest" active loop level cue request must be serviced before a higher level loop request. Obviously, if the inner loop has not become active, as is the case for 1/4 second above, the upper level request is active.

B. Run-Time Variables

The second most important advance in multi-image programming in the AMPL/M system is the introduction of real run-time "variables". A variable in AMPL/M is exactly the same as a variable in BASIC, FORTRAN, etc., for those of you familiar with other computer languages (AMPL/M is the first true multi-image programming language, in the full sense of the word). For those of you not familiar with computer languages, the variable is something you learned about in high-school

algebra. It represents a value which is not known, or can be computed from a "formula", such as "X = X + 4". Don't confuse the variable concept with the syncrolink "adjustable" value. By adjustable, we mean that the value can be adjusted during the programming phase of production by using the syncrolink feature of the AMPL/M editor. A variable is computed during the actual run of a show, by using formulas supplied by the producer during the programming phase.

The "M" version of AMPL has two variable "types", X and C. The "X" type variables are relative time variables, and can have a value from 0.00 to 99.99 seconds. While zero is a legal value, if you try to wait zero seconds, an error message will be given. While the SUPERSTAR may be fast, we haven't invented Faster-Than-Light cues yet!

The "C" type variables are loop count variables, and can have a value from 0 to 9999. A repeat zero count causes a repeat loop to not execute at all, but to simply be skipped.

There are 10 variables of each type, indicated by a single digit following the type character. For example, X1, C4, X0, and C7 are all variables. At the beginning of a program, all variables are undefined, rather than defined as zero. Thus, if you use a variable before defining its value with a formula, than an error message will result.

The X and C values are computed with formulas supplied in LET statements. The LET statement can be used to set an initial value, or to compute a new value. All of the possible formats for the LET statement are shown in the table below. The letters i, j, and k are used to represent digits for the variable "names".

formula	user types	function
LET Ci = nnnn	LCi=nnnn	initialize Ci to nnnn
LET Ci = Cj	LCi=Cj	set Ci to value of Cj
LET Ci = Cj + nnnn	LCi=Cj+nnnn	set Ci to value of Cj plus a constant nnnn
LET Ci = Cj - nnnn	LCi=Cj-nnnn	set Ci to value of Cj minus a constant nnnn
LET Ci = Cj + Ck	LCi=Cj+Ck	set Ci to sum of values of Cj and Ck
LET Ci = Cj - Ck	LCi=Cj-Ck	set Ci to difference of values of Cj and Ck

The same formulas may be used with X values. Simply replace each C with an X in the table above. Note that, while i, j, and k represent three different digits, they all can have the same value. Thus, to double the value of a variable, you can use the statement "LET C1 = C1 + C1". Since only addition and subtraction are allowed in this version, multiplication, percent, division, or geometric progressions must be calculated using multiple additions and subtractions. Also, since only positive values of X and C variables are supported, any negative result of a subtraction will cause an error message to be displayed.

Now that we have discussed what the variable is, and how formulas can be used to compute values, lets look at the applications for this

new AMPL/M feature. Once you get the idea, you will see why we are so excited about the variable concept in multi-image programming! Look at the following example:

```
NO-ADVANCE 2
WAIT 0.10
REPEAT 10
 LET X1 = 0.02
 REPEAT 20
  REPEAT 3
   CUT 2
   WAIT X1
  END: REPEAT
  WAIT X1
  LET X1 = X1 + .01
 END: REPEAT
 SOFT-CUT CUR 2
 WAIT 0.50
 ADVANCE (DARK) 2
 WAIT 1.00
END: REPEAT
```

This example shows how to do a "slow-down" effect. The innermost loop does a triple cut on dissolve unit 2, at a rate determined by the variable X1, initially set to 0.02 (this is 50 cuts per second). The next level loop recomputes the value of X1 by adding 0.01 seconds to it on each repeat, and waits X1 seconds again. This doubles the time between triple cuts. Since this loop repeats 20 times, the triple cut will repeat 20 times, from an initial speed of 50/sec to a final speed of 4.5/sec (0.22 sec wait). At that time, the loop terminates, and the outermost loop takes over. It fades the current lamp from the screen with a SOFT-CUT, waits for the image to fade (WAIT 0.50), advances all three projectors, and waits until the advance cycle completes. Then the entire process is started again. The outer loop repeats 10 times, so the slow-down animation will run through 10 sets to slides. This program is included on your AMPL/M Master Diskette as "SLOW-DOWN DEMO".

Obviously, the LET statement can also be used for speed-up effects as well. Also, by using two variables, you can get other rates of speed-up or slow-down, besides the linear effect in the example above.

Lets look at several more examples of the use of program variables:

```
LET C1 = 1
REPEAT 20
 REPEAT C1
  CUT 1
  WAIT 0.1
 END: REPEAT
 WAIT 0.25
END: REPEAT
```

This example illustrates how the C-type variable can be used to change the repeat count of a loop. In this example, the inner loop will repeat longer and longer on each run. The first time, it will

repeat once. The second time through the outer loop, it will repeat twice, etc. You can combine this increase of repeats with a speed-up effect, so the total time of each inner loop will take the same amount of time! A speed-up or slow-down effect with this length adjustment is really quite interesting! Look at the following example:

```
LET C1 = 1
LET X1 = 0.12
REPEAT 10
 REPEAT C1
  REPEAT 3
   CUT 2
   WAIT X1
  END: REPEAT
 END: REPEAT
 WAIT 0.50
 WAIT X1
 LET X1 = X1 - 0.01
 LET C1 = C1 + 1
END: REPEAT
```

This example is also included on your System Master diskette as "VARIABLE LOOP DEMO". Why not run it now, to see how it works?

Another popular example is the ANIMATE-FADE effect, made popular by one of our competitors. At one point in our history, it was virtually the only thing they could do that we couldn't, and BOY! Did they sell it! Well, now we can do it too! Actually, we have always been able to do it, but not easily. The program variables makes it easy to do:

```
NO-ADVANCE 1
WAIT 0.02
LET X1 = 0.02
LET X2 = 0.18
REPEAT 18
 SOFT-CUT 1
 WAIT X1
 HOLD CUR 1
 WAIT X2
 LET X1 = X1 + 0.01
 LET X2 = X2 - 0.01
END: REPEAT
```

The above example does an animate-fade UP. To go DOWN, exchange the WAIT X1 and WAIT X2 statements. To make a longer fade, put a repeat loop around the first four cues in the loop. To make a shorter fade, use increments of 0.02, and reduce the repeat count to 9. Combine both methods to get rates in between, etc., etc.

C. Multiple Time Lines: The Task

Certainly the most significant advance in multi-image since syncrolink was introduced in 1978 is the MULTI-TASKING ability of the SUPERSTAR. There are several imitations of this feature available with other programming systems: "loop loads", "preprogrammed animation effects", etc., have been offered by several manufacturers

for some time now. The major limitation with these features is that they can only do exactly what the manufacturer designed them to do: specific animation effects. Because they are not really a general tool for multiple simultaneous effects, you quickly exhaust the possibilities with them.

To overcome this, the AMPL/M system allows you to run up to four completely independent programs at the same time. And each program can be as complex or as long as you want! There are no limitations other than the power of the AMPL/M language itself (yes, AMPL/M is limited - It can't make coffee, can it?). The implications of this are far reaching, and will impact multi-image software in the years to come in some pretty amazing ways!

Frankly, we don't really know all the possibilities ourselves! We invented the feature because of its conceptual capability to solve many typical multi-image problems (MULTI-image, by its very nature, demands MULTI-programming). This section is designed to cover the basics of multi-tasking, but the real creativity we'll have to leave up to you! A year or so from now, we can write a really good manual on multi-tasking, once you creative producers teach us how to use it!

The basic idea is quite simple. Any program section can be made into a "task" by enclosing it between a TASK and an END: TASK statement. These two statements work just like a repeat loop, in that the task statements are indented one space. Look at the following example:

```
TASK: ANIMATE-FADE-1
 NOTE: SCREEN 1 MUST BE IN N/A
 LET X1 = 0.02
 LET X2 = 0.18
 REPEAT 18
  SOFT-CUT 1
  WAIT X1
  HOLD CUR 1
  WAIT X2
  LET X1 = X1 + 0.01
  LET X2 = X2 - 0.01
 END: REPEAT
END: TASK
```

Since the maximum indent level allowed in AMPL/M is 4, only 3 levels of nested repeat loops are allowed inside a task. Also, task "blocks" cannot be nested.

Once a section of program has been made into a task, it must be moved to the end of the program, using the block replicate and block delete editor commands. If the SUPERSTAR runs into a TASK statement while running a program, it will assume that the end of program has been reached. So how do you use tasks, if you can't just run them?

I'm glad you asked! AMPL/M has a special statement for this purpose: START TASK: Task-name. This statement runs the named task for you, but also continues running the program as well. That is, the START TASK statement operates like a LET statement in the main program, in that it takes zero program time to execute. The program continues on, just as if the statement wasn't there, except now

another program is running at the same time! Think of it like four little SUPERSTAR computers inside your SUPERSTAR! Each one can be running a completely separate program! They don't even have to be completely separate: in fact they normally will be part of one larger program.

This makes the task work like a "cue file", with the major exception that you don't have to put the cues in the program at every point you want to use them, and that three different "cue files" can be running while the "main" program is running. For example, to do a animate-fade between two sets of slides, use the following sequence:

```
START TASK: A-F-DOWN-2
START TASK: A-F-UP-1
```

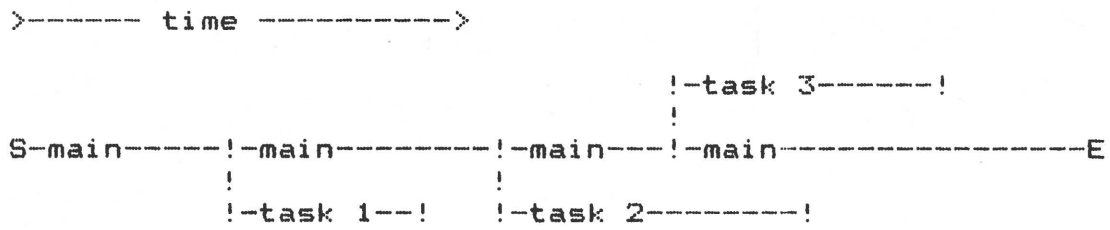
This program sequence assumes that you have two tasks named "A-F-DOWN-2" and "A-F-UP-1". The first will animate-fade down the three slides on screen two, while the second will animate-fade up the three slides on screen 1! And, while this is happening, your "main" program can be chugging along, synchronized to the soundtrack, doing some other effects!

The SUPERSTAR "merges" the different "time-lines" that are running, rather than forcing you to merge them by hand. This makes repeat loops and syncrolink much more useful in complex shows. You can have syncrolinked effects running simultaneously with animation effects! No longer do you have to spread out a simple repeat loop, so you can insert (by hand) other cues to do other effects at the same time!

Obviously, there are restrictions in timing when several time lines are running. The SPEED statement sets how fast the SUPERSTAR can "ship" commands to the dissolves. This sets up shipment "time-slots", which can be filled by any time line. If two separate time-lines want to ship a command at the same time, the commands are "buffered" by priority. The main time line has highest priority, the first task started has second priority, etc. Up to 4 commands can be buffered at any given instant, which means the last one gets delayed 4 time-slots. At 100 cues per second, this is not noticeable. At 10 cues per second, it is quite noticeable. You will find that multi-tasking is most useful at 50 or 100 cues per second, for this reason.

If the shipment buffer gets more than 4 commands in it, a "SHIP BUFFER OVERFLOW" error will occur, and the program will stop running. Keep in mind that 4 time lines with 100 cue per second capability does not mean you can run at 400 cues per second! With four time lines, the average cue speed per time line must not exceed 25 cues per second.

Remember the cue requests discussed previously? Well, with several time lines running, it is possible to have cue requests coming from each one of them! This can be reduced by good programming techniques, such as syncrolinking all sequences as part of the main program first, and then making it a task. For example, lets say your final program will consist of a main program and three different tasks, as diagrammed below:



The "S" indicates the start of the program, and the "E" indicates the end. Each task is diagrammed running parallel to the main program during the time it is running. Note that this example only uses 3 time lines at once.

When programming this show, program down to the point where the first task is to be done. Program the first task on the main time line, and use syncrolink, etc. When you are satisfied with the effect, make a task block out of it. No need to move it, because it is already at the end of the program at this point.

Now go back to the beginning of the task, and insert the next section of the main program. Test and debug it, up to the point of the second task. Now program the second task as part of the main program, as before. Make it a task block, as before, etc. Continue this process to the end of the program. This basic approach will simplify your thinking as you proceed with the programming of a multi-task program.

Once you program is completed, you may still want to go back and modify the timing of a task with syncrolink. This is possible using the "on-the-fly" block syncrolink function. Normally, the syncrolink function will only work on the main time line. However, if you put the block marks inside a task block, the SUPERSTAR will syncrolink there instead. Also, the CRT display will normally follow the main time line, but if syncrolink is active on a task time line, the CRT display will show that time line.

The UNTIL Cn = 0 statement was designed to allow you to synchronize effects between time lines. For example, if you have two different animation effects running, and you want one manual cue to terminate them both, don't use two REPEAT UNTIL CUE loops! They will require TWO manual cues to terminate the two separate loops! Rather, let one loop use the UNTIL CUE statement, and when it exits, set a count value to 0. The other loop should exit with a UNTIL Cn statement. Look at the example loops below:

```
time line #1

NOTE: SET C8
LET C8 = 1
REPEAT
 UNTIL CUE
 CUT 1
 WAIT 0.20
END: REPEAT
NOTE: CLEAR C8
LET C8 = 0
```

```
time line #2

REPEAT
 UNTIL C8 = 0
 WAIT 0.10
 CUT 2
 WAIT 0.10
 CUT 3
END: REPEAT
```

The effect on time line #1 makes the manual cue request, and sets up C8 for time line #2. Time line #2 animates until time line #1 tells it to stop, via variable C8. Amazing!

Autosync timing ambiguities can occur with multiple time lines. In section VI we discussed this problem for single time lines. The potential confusion can get much worse when several time lines are involved. When the computer autosyncs to a particular cue on the main time line (you cannot autosync to a cue inside a task), it will substitute the shortest possible time (set by the speed statement) for all unknown wait values or manual cues. This will usually always shorten up a sequence. This may cause the screen display of images after autosync to be different than you expect! The only way to avoid this is to use lots of WAIT UNTIL T statements, and as few manually cued effects as possible, especially when several time lines are active.

Even though this timing ambiguity exists when autosyncing, no such ambiguity exists when you run the show from the beginning. And if the program is completely canned (no manual cues required at all), then there will be no autosync ambiguity, either!

That just about covers the basics on multi-tasking (I hope!). I'm sure you will have fun working with it, and will get some surprising results which you may not be able to understand. Keep working on it! Once you master the use of this capability, you will be able to produce effects never before practical or possible!

APPENDIX I

AMPL/M STATEMENTS SUMMARY

Each AMPL/M statement is constructed from a series of fields. A field requires certain input from the keyboard, and responds with specific character displays on the CRT. Back-arrow editing is allowed within a field, but once the field is terminated by the user, no additional edits are possible within the terminated field. Fields are terminated with a FIELD SEPARATOR character. Usually, the field separator character is the first character to be entered into the following field. If an error was made in a previous field, the <CTRL-X> key may be used to start over.

The function of the field construct is to restrict the user input to correct AMPL/M statement syntax during statement entry, and to reduce the amount of typing required to enter a statement. If the user attempts to enter any illegal sequence of characters, the computer immediately responds with a "beep", giving instantaneous feedback. This makes the SUPERSTAR typewriter keyboard entry similar to a dedicated computer entry system.

There are 7 common statement fields, which will be described first. Then each statement will be described, using the common definitions.

A. Common Fields

Fields are defined using a special notation. Angle brackets are used to enclose field names, and curley brackets are used to indicate optional fields or entries. Slashes are used to indicate choices. For example:

<digit> := [0/1/2/3/4/5/6/7/8/9]

The above definition states that a digit field consists of one of the 10 decimal digits. Seems simple enough, right? The ":=" means "is defined as". For review, here is a list of all the special characters used in the statement definitions:

character	meaning
/	"or" (choice)
{}	optional item
<>	name of item
[]	contains choices
:=	"is defined as"

Next, lets define the 7 common field types. Rather than try to define them with the above notation, the basic definitions will be made in english.

1. <statement-name>: This field begins every AMPL/M statement, and is entered with two characters. In special cases, the second character can be a <SPACE>. The legal codes are

given in the AMPL/M statements menu, and with each statement in this appendix. The computer will type the entire statement name on the CRT screen when the second character is entered. Backarrow editing is allowed. Any third character (other than the backarrow) is a field separator, and causes exit to the next field.

2. <access>: This optional field allows the user to specify projector access. The field is not entered if the separator character typed in the statement-name field was a decimal digit.

   The access type allowed is determined by the editor access mode, controlled by the <CTRL-A> key. The access mode can be toggled at any point in the statement except within this field. If absolute access is selected, then the letters A, B, and C may be used as entry. Any combination is allowed, but they must be entered in the A-B-C order.

   If relative access is selected, then the input must be selected from C, N, and T. These letters are expanded to CUR, NXT, and THD on the screen, to improve readability. Again, any combination of the letters is allowed, but they must be entered in the C-N-T order.

   Field editing is allowed, using the back-arrow. The access field is terminated when any decimal digit is typed, which is passed on to the following screen area field. Thus, any digit acts as a field separator.

3. <screen-area>: This field specifies the destination dissolves for the specified statement and access. All ten decimal digits may be used, but they must be entered in ascending order. Note that the digit "0" refers to screen 10, so it must be the last screen entered. When more than one digit is entered, commas are automatically inserted between screen digits. If the statement is too long to fit on the screen, the commas are automatically removed. Backarrow editing is allowed, and the field (and statement) is terminated with the <SPACE> or <CUE> separators.

4. <x-value>: This field accepts a wait value from 0.01 to 99.99 seconds. This field is broken up into two separate fields, seconds, and hundredths. The field can be entered with one of the following formats:

   N NN N.N NN.N N.NN NN.NN .N .NN

   If you type a digit as the first character, you must enter a seconds field. Zero is permitted, if you hit a digit by mistake. The decimal point is a field separator. Once you hit the decimal point, you can't go back and edit the seconds, without using <CTRL-X>. The field (and statement) are terminated with a <SPACE> or <CUE>.

5. <t-value>: This field accepts an absolute time track value. The hours, minutes, and seconds are separate fields. One digit must be entered for hours, 0, 1, or 2.

You may back up to change this digit, or type a <SPACE> or colon (:) to enter the minutes field. The <SPACE> and colon are both legal field separators.

In the minutes field, one or two digits are allowed, from 0 to 59 minutes. Again, you can use the backarrow to edit this field. Press <SPACE> or colon to go to the seconds field. If you have entered a value greater than 59 minutes, or if you have specified more than 2 hours, 39 minutes, an error will be given at this point. The seconds field works exactly the same as the <x-value> specified above, except you can only enter up to 59.99 seconds.

6. <c-value>: This field allows you to enter a count value from 1 to 9999. Backarrow editing is allowed. A <SPACE> or <CUE> terminates the field (and the statement).

7. <name>: This field is used for tab and task names. The name may contain up to 18 characters, and must start with an alphabetic character. The remaining characters may be any printable character, except a <SPACE>. The <SPACE> or <CUE> terminates the field (and the statement).

B. Statement Definitions

Below, each AMPL/M statement is defined, using the above definitions. The underscore character (_) is used to indicate <SPACE> on code inputs. All statements are terminated with either a <SPACE> or <CUE>. For more details on how to use these statements, refer to sections XIX and XX of this manual.

1. ABSOLUTE-ACCESS <screen-area>

   Example: ABSOLUTE-ACCESS 1,2,3

   Code: AA

   Description: Toggles the access mode of the selected dissolve between absolute and relative. Current dissolve mode is displayed in the status window.

2. ADVANCE {<access>} <screen-area>

   Example: ADVANCE CUR 1,2,3

   Code: AD or A_

   Description: Causes the selected projectors on the selected screens to advance one slide position. Tray position is available from the editor command mode by typing <X>.

3. AUXILIARY <screen-area>

   Example: AUXILIARY 1,3,5

   Code: AU

Description: Operates the built-in auxiliaries of the Star System.

4. CUT {<access>} <screen-area>

   Example: CUT A,B 3,4,5

   Code: CU or C_

   Description: This statement does an instantaneous dissolve on the selected projectors.

5. END: REPEAT

   Example: END: REPEAT

   Code: ER or E_

   Description: Used to terminate a repeat loop.

6. END: TASK

   Example: END: TASK

   Code: ET

   Description: Used to terminate a task.

7. EXTERNAL [(PULSE)/(ON)/(OFF)] <BUC>

   Example: EXTERNAL (PULSE) 125

   Code: EX

   Description: The first field of this statement can be (PULSE), (ON), or (OFF). These three choices are entered with P, N, and F, respectively. The BUC code specifies the external auxiliary to be accessed, and must be three digits, The maximum value of each digit is 2, 4, and 7, respectively. A zero digit is not allowed.

8. FREEZE {<access>} <screen-area>

   Example: FREEZE NXT 8

   Code: FR or F_

   Description: Removes projectors from the automatic sequence in the dissolve. Also can be used to set the Star-3 Dissolve for 2-projector operation.

9. HOLD {<access>} <screen-area>

   Example: HOLD CUR,THD 1,2,3,4,5

   Code: HO or H_

Description: Stops or releases the fader function in the selected projectors.

10. LET C <expression>

Example: LET C1 = C1 - 1

Code: LC

Description: Used to compute a new value for a C-type variable. Allowed expressions are shown below, with i, j, and k representing single digits, and nnnn representing a C-type value:

statement format	user enters
LET Ci = <c-value>	LCi=nnnn
LET Ci = Cj	LCi=Cj
LET Ci = Cj + <c-value>	LCi=Cj+nnnn
LET Cn = Cj - <c-value>	LCi=Cj-nnnn
LET Cn = Cj + Ck	LCi=Cj+Ck
LET Cn = Cj - Ck	LCi=Cj-Ck

The equal sign, plus sign, and minus sign are all field separators. Each digit or value field may be edited with the backarrow before the field separator is typed.

11. LET X <expression>

Example: LET X2 = X4 + X6

Code: LX or L_

Description: Used to compute a new value for an X-type variable. Works the same as the LET C statement, except all C's are replaced with X's, and the C-value is replaced with an X-value.

12. NO-ADVANCE <screen-area>

Example: NO-ADVANCE 1,8,0

Code: NA

Description: Toggles the no-advance mode in the selected dissolves, and turns off no-delay mode, if active.

13. NO-DELAY <screen-area>

Example: NO-DELAY 1

Code: ND

Description: Toggles the no-delay mode in the selected dissolves, and turns off no-advance mode, if active.

14. NOTE: <comment>

Example: NOTE: THIS IS A SAMPLE NOTE.

Code: NO or N_

Description: Allows entry of up to 18 characters for the purpose of documenting a program.

15. REPEAT CnREPEAT C {=<C-value>}

Examples: REPEAT C3
REPEAT C
REPEAT C = 23

Code: RC

Description: This code will generate two completely different types of REPEAT statements. The first type is a REPEAT loop with the number of repeats specified with a C-type variable, Cn. The digit field can be edited with the backarrow.

The second type of REPEAT C statement is used for syncrolinking the number of repeats. An initial value may be entered. The equal sign is the field separator in this case.

16. REMOTE: [ON/OFF]

Example: REMOTE: ON

Code: RM

Description: Used to turn on or off the remote cue input. The ON or OFF field is entered with N or F, respectively. The field can be edited with the backarrow.

17. REPEAT {<C-value>}

Examples: REPEAT
REPEAT 98

Code: RP or R_

Description: This REPEAT statement also has two forms, since the value is optional. If no value is entered, then the loop must be terminated with a UNTIL statement inside the loop. If no UNTIL statement is found in the loop, the END: REPEAT statement will generate an error message.

18. REVERSE {<access>} <screen-area>

Example: REVERSE A 1,2

Code: RV

Description: Reverses the selected projectors in the selected dissolves.

19. SOFT-CUT {<access>} <screen-area>

    Example: SOFT-CUT CUR,NXT 1

    Code: SC or S_

    Description: Causes a 1/4 second fade on the selected projectors.

20. S/HOME <screen-area>

    Example: S/HOME 1,2,3,4,5,6,7,8,9,0

    Code: SH

    Description: The first execution of this statement on a particular dissolve will turn off all lamps, and reset the mode to standby, normal advance, relative access, and 3-projector mode. If this statement is executed twice to the same dissolve, with no other dissolve statement in between, then the slide trays will be homed. S/HOME stands for "standby/home".

21. SPEED: <rate>

    Example: SPEED: 100/SEC

    Code: SP

    Description: Sets the minimum wait time between cues shipped to the external interface (Star-3 Programmer, Universal Interface, or Star Controller). Legal values are 10 CUES/SEC, 50 CUES/SEC, and 100 CUES/SEC, entered with the single characters 1, 5, and 0, respectively. Rate field can be edited using the backarrow.

22. START TASK: <name>

    Example: START TASK: FLICKERING_FLAME

    Code: ST

    Description: Initializes a new time line, and begins the named task running on the new time line.

23. TAB: {<name>}

    Example: TAB: CREDITS_SEQUENCE

    Code: TB or T_

    Description: Allows a permanent tab to be placed in the program. The tab can be named or unnamed, depending on the intended usage.

24. TASK: <name>

Example: TASK: BOUNCING_BALL

Code: TK

Description: Defines the beginning of a task block.

25. UNTIL C<digit> = 0

Example: UNTIL C3 = 0

Code: UC

Description: Used to test a C-type variable for a value of zero, to cause a loop termination. Purpose of this statement is to allow synchronization of different tasks. The only entry required by the user is a single digit, followed by a <SPACE> or <CUE> to terminate the statement. Backarrow editing of the digit is allowed.

26. UNTIL LAST

Example: UNTIL LAST

Code: UL

Description: Causes exit from the loop if the loop counter equals the maximum value set at entry by the REPEAT statement. Will only work with loops with fixed repeat counts (REPEAT nnnn, REPEAT Cn, or unsyncrolinked REPEAT C = nnnn).

27. UNTIL CUE

Example: UNTIL CUE

Code: UQ or U_

Description: Causes exit from a loop if a manual cue is received during the execution of the loop.

28. WAIT <X-value>

Example: WAIT 2.13

Code: WA or W_

Description: Allows a fixed delay between cues, from 0.01 to 99.99 seconds.

29. WAIT UNTIL T {=<T-value>}

Example: WAIT UNTIL T
WAIT UNTIL T = 0:05:15.00

Code: WT

Description: Sets up a syncrolink type absolute wait statement. An initial value may be entered with the

statement.

30. WAIT X {= <X-value>}

Example: WAIT X
WAIT X = 1.24

Code: WX

Description: Sets up a syncrolink type wait statement. An initial value may be entered with the statement.

31. NN-SEC {<access>} <screen-area>

Example: 1-SEC CUR,NXT 1,3,5
12-SEC 8
4-SEC A,B 3,9

Code: see table below:

statement	codes
1-SEC	1S or 1_
2-SEC	2S or 2_
4-SEC	4S or 4_
6-SEC	6S or 6_
8-SEC	8S or 8_
12-SEC	12
16-SEC	16
24-SEC	24

Description: The selected projectors on the selected dissolves are faded at the selected rate.