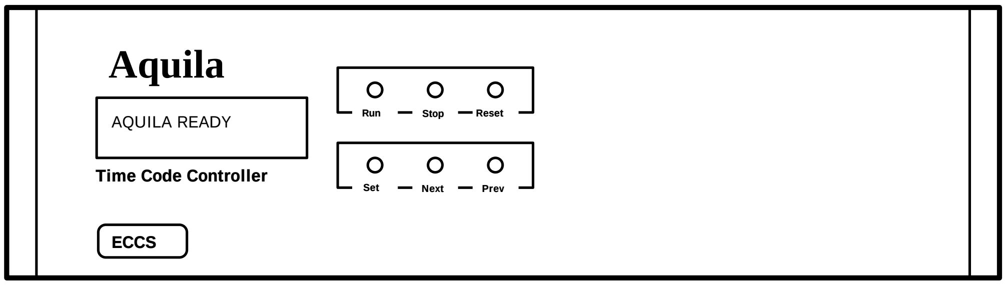
The Aquila Time Code Controller



Software Version 1.99

10/12/01

East Coast Control Systems, Inc.

347 Bigler Road, P.O. Box 486

Bigler, PA 16825

(814) 857-5420

Table Of Contents

The *AQUILA* Time Code Controller

General Description

What follows is a brief overview of the functions and capabilities of the *Aquila Time Code Controller*. More detailed information may be found elsewhere in this manual.

The *Aquila* is a general purpose time code box capable of reading and writing many industry standard time codes. During time code read, extensive tests are performed to insure the validity of the code that is displayed and reported. During fault conditions, the *Aquila* can 'freewheel', maintaining a continuing time code stream.

Continuity between time frames is also tested. As each time frame is successfully read, it is checked with the previous frame to make sure that the code is monotonically increasing. Should the current frame appear to be out of sequence, the *Aquila* will report the time as the previous time plus one frame. Upon receiving two consecutive frames that are monotonically increasing, the *Aquila* will accept the new time as valid. The time code can be displayed on the built-in LCD display as well as transmitted to another location or computer host via a standard serial port. Time code can be reported via the serial port continuously, or only when polled. Alternatively, the user may set an 'Alarm'. In this mode of operation, the time is reported only when the time code value reaches the 'Alarm' time. Computer communication data format is in ASCII form, allowing 'live' communication with the unit via any available terminal program on any computer system.

Stand alone operation of *Aquila* mode settings and time code operation are facilitated through convenient front panel controls and a 16 character by 2 line backlighted LCD display. Mode setup and operation can also be accomplished via a host computer.

The *Aquila* also includes eight dry contact switch closures to control external devices. These closures may be instructed to operate in a latched mode or a pulsed mode. They are controlled by means of commands issued from a host computer, or by commands from an internal instruction file.

The connections between the time code box and the source/destination of the time code (usually a tape deck) are electrically isolated. This increases reliability by eliminating common-mode noise problems, and insures that the *Aquila* will not create any grounding problems such as ground loops and hums.

Software modifications to the *Aquila* can be performed in the field. Program upgrades can be supplied on disk and uploaded to the *Aquila* by means of the serial port. In this way, the capabilities of the *Aquila* can grow without downtime or expensive service calls.

The Sequencer operation of the *Aquila* allows time based programs to be written with any word processor and uploaded to the unit. These programs are stored in nonvolatile memory in the *Aquila* for playback at any time. Program execution can be driven either by internally generated time code or external time code, and can occur without host computer interaction. The program length is limited only by the amount of nonvolatile memory available for that purpose, but can easily exceed 1200 timed events.

Several options are available to expand the capabilities of the *Aquila*. The Parallel Expander Board (PEB) provides 32 TTL compatible digital input/output bits (Port A) and 32 TTL compatible digital output only bits (Port B). A host computer can read and/or set these bits at any time. Additionally, the bits can be controlled by means of the *Aquila's* built-in *Sequencer.*

Using the PEB, the *Aquila* can also function as a data or *Sequence* recorder. In the record mode of operation, the *Aquila* monitors 32 bits of information from PEB Port A while time code is running. Whenever any bits change state, the *Aquila* records the time of change as well as the change itself, effectively writing its own instruction file for *Sequencer* operation. The series of bit changes can then be played back, driven by internal or external time code.

Beginning with firmware revision 1.90, an *Aquila* with a PEB can also use the 32 available SMPTE user bits to store and playback bit information. In the *Record to SMPTE* mode, the *Aquila* monitors 32 bits of information from PEB Port A and inserts this information into the SMPTE user bits as it generates a SMPTE time code stream. In the *Playback from SMPTE* mode, the *Aquila* examines the user bits on every valid SMPTE frame read and transfers this information to PEB Port B.

The *Aquila* is available in an attractive black desktop enclosure. The unit can also be packaged in a sturdy 19" standard rack mount enclosure. Two rack units high, this package can provide room for additional interface hardware options, such as up to four of the Octal Relay Boards now available. Used in conjunction with the PEB, the Octal Relay Boards can convert the PEB TTL lines to dry contact switch closures for ease of use.

In application, the *Aquila* can perform time code synchronization for theater control systems such as the Universal Theater Control System (UTCS), also from East Coast Control Systems. Additionally, the *Aquila* is recommended for use with the *Laser Show Designer* series of programs from Pangolin Laser Software. Teaming the *Aquila* with LD provides fast and accurate time code synchronization during laser show playback for snappy performances.

The *Aquila* has seen operation in a multitude of other presentation uses, including automating the operation of laser display beam tables and firing pyrotechnics, all synchronized to a master time code.

Features of the *Basic Aquila* include:

- ability to read and write several types of time code
- one serial port for computer control
- eight dry contact switch closures for user defined purposes
- front panel controls and display for stand alone operation and monitoring
- *Sequencer* operation for automatic control of switch closures
- desktop or rackmount packaging

Features of the *Expanded Aquila 64* include all features of the *Basic Aquila* and...

- 32 digital input/output bits and 32 digital output only bits (TTL compatible)
- data recording and playback of 32 bits via *Keepfile* or SMPTE user bits
- optional relay boards to convert some or all of the digital output bits into dry contact switch closures (rack mount only).

The Front Panel - the LCD Display and Controls

The LCD Display

During time code operations, the front panel LCD display shows current input (read) time, or current output (write) time, whichever is appropriate. This time is displayed on the bottom line of the display along with the type of time code in use and direction of time code (in or out). For example, when reading SMPTE time code at 30 frames per second, the time value can be seen in the leftmost 15 characters, while the 4 rightmost characters will display '>S30', for 'in SMPTE 30'. If this timecode is being written instead of read, the 4 rightmost characters will display 'S30>', for 'SMPTE 30 out'.

The top line of the display is used to present the current status of the *Aquila*. Usually, this area will display the message 'AQUILA READY' to signify normal operation or the successful completion of a command from the serial port or the *Sequencer Keepfile*. Any error message generated by an incorrect serial port or *Sequencer Keepfile* command will also be displayed here.

In addition to the *Aquila* status, the 2 rightmost characters of the top line is used to display the quality of any time code being read. The actual value of the two digits is not significant. What is important is the rate of change of the numbers. A good time code will have few changes, while poor quality time code will cause the digits to increment at a higher rate.

The LCD Display Backlight

Operation of the LCD display light is controlled by means of the 'NEXT' and 'PREV' buttons on the front panel. When the Aquila is in normal mode of operations (not in 'SETTINGS' mode), the backlight may be activated by pressing the 'NEXT' button. Pressing the 'PREV' button will deactivate it.

The operation of LCD display light can also be controlled by means of commands issued from a host computer via the serial port, or by commands issued by the *Sequencer Keepfile*.

The LCD Display Viewing Angle

The viewing angle of the LCD display can be adjusted for optimum visibility. To adjust the viewing angle on the desktop model, open the case by removing the two screws from the front panel. Grasp the plastic bezel and gently remove the entire front panel assembly. The adjustment control can be found inside the case on the left beside the area where the ribbon cable from the display is attached to the board. Adjustment can be made with a small screwdriver.

To adjust the viewing angle of the LCD display in a rack mount model, open the case by removing the top cover. The adjustment control can be found inside the case on the left beside the area where the ribbon cable from the display is attached to the board. Adjustment can be made with a small screwdriver.

Using the Controls to Output Time Code

Time code writing with the *Aquila* can easily be accomplished with the front panel controls. Pressing the 'RUN' button will activate the time code output and start the *Aquila* writing time code. Pressing 'STOP' will halt the time code output, but will not reset the time to 0. (A subsequent pressing of the 'RUN' button will restart the time code at the place it was 'STOPPED'.) Pressing the 'RESET' button will stop the time code and reset the time to 0.

Using the Controls to Set the Personality

The *Aquila* provides a convenient menu driven method of setting operating characteristics, or 'personality', by means of the LCD display and the 'SET', 'NEXT', and 'PREV' buttons. Menu selections may be viewed or altered at any time, except when the *Aquila* is writing time code.

To view or change menu selections, enter SETTINGS MODE by pressing 'SET'. The top line of the display indicates the characteristic under consideration, while the bottom line indicates the current selection for that characteristic. The 'NEXT' and 'PREV' buttons move you through the selections for the current characteristic. To keep a selection, press 'SET'. This will also move you to the next characteristic to be considered.

After all characteristics have been viewed, the *Aquila* will return to the normal operating mode. Any changes made will be maintained until they are changed again, even if the *Aquila* is powered down between uses.

If you press 'RESET' at any time during the menu selection process, the *Aquila* will immediately exit the menu *without altering the current characteristic* and return to normal operations.

The following table lists the characteristics and selections available, and provides a short description of each. Additional information is available throughout this manual.

Locking the Settings

The menu settings can be *locked* so that they can not be accidentally changed. The last menu selection will allow you to lock or unlock the settings.

Aquila Characteristics and Settings

MODE:	CONTINUOUS	Time is reported to the host on every time change.
	POLLED	Time reported upon request. An 'Alarm' may be set.
	EPOCH - ASCII	Time reported on the second in the normal protocol.
	ECCS - UTCS	Time is reported on the second in ECCS protocol.
CLOCK TYPE IN:	BCD10	Binary Coded Decimal - 10 frames per second.
	SMP10	SMPTE - 10 frames per second.
	SMP24	SMPTE - 24 frames per second (motion pictures).
	SMP25	SMPTE - 25 frames per second (European TV).
	SMPDF	SMPTE DROP FRAME - 30 frames/sec (US TV).
	SMP30	SMPTE - 30 frames per second.
	AVL10	Audio Visual Labs proprietary - 10 fps.
CLOCK TYPE OUT:	BCD10	Binary Coded Decimal - 10 frames per second.
	SMP10	SMPTE - 10 frames per second.
	SMP24	SMPTE - 24 frames per second (motion pictures).
	SMP25	SMPTE - 25 frames per second (European TV).
	SMPDF	SMPTE DROP FRAME - 30 frames/sec (US TV).
	SMP30	SMPTE - 30 frames per second.
FREEWHEEL:	NONE	Time code operations stop immediately.
	30 FRAMES	Time code operations 'coast' for this duration
	600 FRAMES	when incoming time code stops or fails.
	INFINITE	Operations continue until 'RESET' is pressed.
REGENERATION:	OFF	Time Code Regeneration/Conversion disabled
	ON	Time Code Regeneration/Conversion enabled
BAUD RATE:	300	
	600	
	1200	
	2400	Lowest baud rate to report every frame at 10 fps.
	4800	
	9600	Lowest baud rate to report every frame at 30 fps.
	19200	
	38400	(Use at own risk. Not certified reliable in V1.xx)
	76800	(use at own risk. Not certified reliable in V1.xx)
COMM ECHO:	OFF	Characters from host not echoed back to host
	ON	Characters from host echoed back to host.
RECORD DATA:	OFF	Data record disabled (PEB only)
	REC TO KEEPFILE	Data record to KEEPFILE (PEB only)
	REC TO SMPTE	Data record to SMPTE user bits (PEB only)
PLAYBACK:	OFF	KEEPFILE or SMPTE playback disabled
	FROM KEEPFILE	Playback from KEEPFILE
	FROM SMPTE	Playback from SMPTE user bits
	FROM SMP+KPFL	Playback from SMPTE user bits and KEEPFILE
AUTO REST:	OFF	AUTO REST disabled
	ON	AUTO REST enabled
LOCK:	OFF	EDIT SETTINGS OK
	ON	SETTINGS LOCKED

Rear Panel Connectors

The *Basic Aquila* has four wedge shaped connectors on the rear panel. All input and output connections are made here. They are called DB connectors, and are commonly found in the computer industry. The two nine pin connectors (DB-9's) are for serial communications with a computer. The 25 pin connector (DB-25) is for the eight dry contact switch closures. The 15 pin connector (DB-15) is called the auxiliary connector.

The *Expanded Aquila 64* has two additional 37 pin wedge shaped connectors (DB37's) to handle the additional 64 bits of control information. Consult the section on the Parallel Expansion Board (PEB) for more information.

Rack mount *Expanded Aquila 64's* may have additional male DB25 connectors on the rear panel. Each DB-25 connector will be for an *Octal Relay Board*, each of which supplies an additional eight dry contact switch closures. Consult the section on the Octal Relay Boards for more information.

The Auxiliary Connector

The female DB-15 connector on the rear panel is known as the *Auxiliary Connector*. Several signals are available here, including the Time Code In and Time Code Out signals. These are balanced, floating, high impedance signals which allow direct connection with most time code sources without worrying about ground loops and hums. The time code pins of interest are listed in the table below. Note that Frame Ground (pin 9) cannot be used as a signal ground. Use the (+) and (-) pins for Time Code In and Time Code Out for proper operation.

Digital Inputs

The *Aquila* also makes available eight TTL and CMOS compatible digital inputs. These inputs are tied to a logic high internally through 10K pull up resistors, and are activated by pulling them to ground with a switch closure, transistor, TTL or CMOS IC, or equivalent. Six of the inputs are used with the front panel buttons, and two are currently undefined in V1.xx software. The table below lists the digital input pin assignments.

External Sync

The Auxiliary Connector also makes available an 'External Sync' input. The software to handle this signal is not implemented in V1.xx software.

DB-15 Auxiliary Connector Pinout

1	External Sync		9	Ground	
2	Time Code In (+)		10	Time Code In (-)	
3	Time Code Out (+)		11	Time Code Out (-)	
4	+5V Out		12	PB0	'RUN'
5	PB1	'STOP'	13	PB2	'RESET'
6	PB3	'SET'	14	PB4	'NEXT'
7	PB5	'PREV'	15	PB6	unassigned
8	PB7	unassigned			

Switch Closure Connector

The male DB-25 connector on the rear of the *Aquila* provides eight dry contact switch closure outputs for user defined applications. Normally Open and Normally Closed contacts are available for each of the eight switches. Each switch is capable of carrying up to 1/2 amp of a resistive load. The pin assignment for this connector is listed in the following table.

These switches can be operated by commands sent by a computer via the serial port, or by commands issued by a *Sequencer Keepfile*.

DB-25 Switch Closure Connector Pinout

1	NC1	14	NO1
2	C1	15	NC2
3	NO2	16	C2
4	NC3	17	NO3
5	C3	18	NC4
6	NO4	19	C4
7	NC5	20	NO5
8	C5	21	NC6
9	NO6	22	C6
10	NC7	23	NO7
11	C7	24	NC8
12	NO8	25	C8
13	Frame Ground		

Data Communications Connectors

A male and a female DB-9 connector are located on the *Aquila* rear panel to handle communication with a host computer or control system. The two connectors are wired together internally and are provided in order to allow the *Aquila* to easily connect to a number of devices that may need to receive time code information. The table below lists the pins and signals on these connectors. The Appendix contains diagrams to illustrate how to correctly wire a cable that will allow communication between the *Aquila* and several common computers.

DB-9 Communication Connector Pinout

1	TxD	Transmit Data	6	na	
2	/TxD	Transmit Data (inv)	7	RxD	Receive Data
3	/RxD	Receive Data (inv)	8	na	
4	na		9	na	
5	GND	Frame Ground			

Time Code Operations

The *Aquila* is simple to operate. The most commonly used functions are accessible from the front panel to facilitate stand-alone operation, while all functions can be activated from a host computer attached to the serial port.

Time Code Cable Hookup

The time code cable supplied by ECCS consists of a DB-15 connector attached to a pair of RCA audio connectors. The DB-15 attaches to the Auxiliary connector on the rear of the *Aquila*, while the RCA connectors attach to the time code source and/or destination. The red RCA connector provides the output time code signal from the *Aquila* and should be attached to the time code destination (usually the input of a tape deck). The black RCA connector supplies the input time code signal to the *Aquila* and should be attached to the time code source (usually the output of a tape deck).

Alternatively, the user may supply an appropriate time code cable for use with the *Aquila.* Refer to section 3 of this manual for more information.

Placing Time Code On the Tape

Once the *Aquila* is attached to the tape deck, select the type of time code to place on the tape by means of the front panel menu controls (see section 2) or host computer commands. Unless the application specifically calls for something else, the usual time code selection is SMPTE 30.

Once the appropriate time code selection has been made, put the tape deck into the 'Record', 'Pause' mode and start the *Aquila* writing time code by pressing the 'RUN' button (or use the appropriate computer command). Set the record level on the tape deck to a suggested 0db. (The unit should function reliably on any signal between -20db and +5db.)

After the tape deck input level is adjusted, press 'RESET' on the *Aquila* to reset the output clock to 0, start the tape deck in 'Record' mode, and once again press 'RUN' on the *Aquila*. The *Aquila* will automatically display the outgoing time on the LCD readout. When the tape has been 'striped', stop the deck and 'RESET' the *Aquila.*

Reading Time Code

To read the incoming time code, connect the *Aquila's* Time Code In line to the source of time code. Set the type of time code to be read by means of the front panel or the appropriate computer commands.

Important! The *Aquila* must be set to read the correct type of time code. Failure to match the *Aquila* CLOCK TYPE IN setting to the actual type of time code expected may result in erratic operation or failure to read time code.

The *Aquila* is always ready to read (unless it's writing), and will start reading the time code as soon as it begins. The *Aquila* will automatically display the incoming time on the LCD readout.

Freewheeling

In order to prevent the false triggering of events, the *Aquila* constantly tests the quality of the incoming time code stream. Should the time code data not pass every test, the *Aquila* enters a fault tolerant mode called 'freewheeling', which allows the *Aquila* (and the presentation) to 'coast' past the time code fault.

The length of time that the freewheeling will occur is adjustable from the front panel menu or from serial port commands. The selections are as follows:

NO FREEWHEEL	Use when the presentation must stop with the time code.
30 FRAMES	Recommended for most applications.
600 FRAMES	Use when time code dropouts exceed 30 frames.
INFINITE	Use when time code is very dirty. Only 'RESET' will stop it.

Time Offsets

The *Aquila* can add or subtract a given offset to the time code that is being read. This offset may be temporary, or can be stored in nonvolatile memory for 'permanent' installations. These time offsets cannot be changed by means of the front panel menu. Only software commands from the serial port or the *Sequencer Keepfile* may affect them.

Note: Only time code reading is affected by time offsets. Time code writing is never affected.

Time Code Regeneration and Conversion

The *Aquila* can convert one type of time code (being read) to another type of time code (being written). This conversion is synchronized to maintain presentation timing.

This capability is available from the front panel menu selections. Set the CLOCK TYPE IN to match the time code source. Set the CLOCK TYPE OUT to match the desired time code. Set REGENERATION to ON.

If you are using the ECCS supplied time code connector, connect the black RCA plug to the time code source and the red RCA plug to the time code destination. The *Aquila* is now ready to perform time code conversions, and will begin as soon as the input time code starts.

If the CLOCK TYPE IN and CLOCK TYPE OUT are set to the same type of time code, the *Aquila* will perform a *regeneration* of the original signal.

If time offsets are programmed into the *Aquila,* the output time code will be offset from the input time code by the offset amount.

Using Serial Communications

The built-in serial communications port makes it possible for the *Aquila* to report the current time code value to a remote location or host computer. Also, it enables the operation of the *Aquila* to be controlled remotely.

Hardware and Software Setup

Connect the *Aquila* and computer with the appropriate serial communication cable. These are available from ECCS directly, or make your own using information supplied in the Appendix.

From the front panel menu, select a *baud rate*, or communications speed. A baud rate of 2400 is recommended for use for *Keepfile* programming and editing operations. All other operations may use baud rates of up to 19200. *Note: The baud rate of the Aquila and the computer must match*.

If you will be communicating via a terminal program, set the *Aquila* COMMUNICATION ECHO ON from the front panel menu. Otherwise, you will (probably) want to turn the COMMUNICATION ECHO OFF.

Set your computer software for '8 data bits, no parity, one stop bit' and turn all handshake signals off. *Aquila* communications should now be established.

To test the communications link, you can type 'VERS' and press the 'enter' key (abbreviated as <enter>). If all is well, the *Aquila* will return the current software version and a copyright message.

Appendix Table A lists all of the commands that the *Aquila* understands, along with a short description of them. The commands are described in detail in Section 6.

Communication Hardware Details

Note: This information is supplied to those users who need more details about the Aquila hardware involved with communications. Nontechnical users may wish to skip this section.

The native hardware protocol for the *Aquila* communications system is a full duplex RS-422/485 standard, with no hardware handshake. This provides the ability to communicate data at high baud rates over relatively long distances. This is also directly compatible with Apple MacIntosh and Apple IIgs serial ports. By means of a specially wired cable, the *Aquila* can also communicate with RS-232 devices such as serial ports found on most PC compatibles and Amiga computers.

RS-422/485 Operation

The RS-422/485 serial communication standard allows connection of several devices simultaneously over a line length of up to 4000 feet at a data transfer rate that can exceed 1 MegaBaud.

When operating the *Aquila* in the RS-422/485 protocol over distances exceeding a few yards, care should be taken to follow the cable recommendations of that standard. Specifically, the cable should contain two individually shielded twisted pairs of wire, with a characteristic impedance of 100 ohms, such as Belden 8102, 8132, or equivalent. Any Category 5 wire is also acceptable.

RS-232 Operation

The *Aquila* can also be attached to serial devices that conform to the RS-232 standard. Appendix A3.1 illustrates the connection that effectively transforms the RS-422 protocol of the *Aquila* to RS-423, which is generally compatible with RS-232. In this configuration, the *Aquila* can be connected to only one device. The Appendix contains drawings for several common cable configurations.

Note: When the Aquila is communicating with RS-232 devices, it is imperative that a ground reference be supplied. Using a common grounded power source for the computer and the Aquila is usually sufficient. If this is not convenient, an extra wire in the communications cable must be used to supply the ground reference. Consult the wiring diagrams in the Appendix for more details.

Aquila Software Control

The *Aquila* is an intelligent, programmable device with many capabilities accessible by means of software control. Except where noted, software commands can be directed to the *Aquila* using any of the following methods.

A computer *terminal program* such as *ProComm* may be used to establish 'live' communication with the *Aquila* by means of the serial communications port. Any command or inquiry directed to the *Aquila* in this fashion will be processed immediately. Most *Keepfile* programming is performed with the aid of a terminal program.

Alternatively, advanced users may want to write a special purpose computer program to control and communicate with the *Aquila* by means of the serial communications port. These programs may typically control a presentation or a process, using the *Aquila* as a timing and/or control element. The *Universal Theater Control System* from ECCS utilizes the *Aquila* in this manner, as does the LD software from Pangolin Laser Software.

Software commands embedded into a *Sequencer Keepfile* can also be used to control the *Aquila*. Except where noted, any command can be included in the *Keepfile* for synchronized playback. Consult the section on *Sequencer* operation for more details.

Time Code Generating Commands

All aspects of time code generating can be controlled by means of software commands. The commands involved in writing time code are as follows:

WRIT		Start writing time code from current time out value.
WSTP		Stop writing time code without resetting time out value.
WRST		Stop writing time code and reset time out value.
TIME	**hh:mm:ss:ff***	Set time out value for writing time code.

*Note: hh = hour value; mm = minute value; ss = second value; ff = frame value.

None of these commands will elicit a response from the *Aquila*, unless an error in the command prompts a 'Syntax Error'.

Reading and Reporting Time Code

The *Aquila* is always ready to read time code unless it is writing code or in the settings mode. When displaying and reporting time, the *Aquila* will use the time value from the input time buffer, unless the unit is writing time code, in which case the output time is used.

Continuous Reporting

If the CONTINUOUS MODE of operation is selected, no commands need to be issued. The *Aquila* will automatically send the time whenever the time code value changes. This time will be an ASCII string in the following form:

Thh:mm:ss:ff<CR><LF>

Note: hh = hour value; mm = minute value; ss = second value; ff = frame value.

It is important to note that the highest resolution of time that can be reported to the host computer in the CONTINUOUS MODE is dependent upon the communications baud rate that is selected. For example, if every frame of a 30 frames per second time code must be captured by the host, the minimum baud rate is 9600. By comparison, if every frame of a 10 frames per second time code must be captured, the minimum baud rate drops to 2400.

Should it be necessary to operate at a baud rate that is slower than the aforementioned minimums, the *Aquila* will report time to the host as many times as it can. For example, at 300 baud, the *Aquila* can report the time approximately twice a second. The time reported is accurate, but intervening frames are not transmitted.

Polled Reporting

In the POLLED MODE of operation, reception of the time value is under control of the host computer. The *Aquila* will not issue a time code value until it is asked to do so. The command to POLL the *Aquila* is as follows:

TIME	Time inquiry (polled mode).

Upon reception of that command, the *Aquila* will immediately respond with the current time value (if time code is running) in the following format:

Thh:mm:ss:ff<CR><LF>

Note: hh = hour value; mm = minute value; ss = second value; ff = frame value.

If there is no active time code, the *Aquila* will reply as follows

M28'CLOCK STOPPED'<CR>

Report On Alarm

From the POLLED MODE of operation, the host computer can set an *alarm*, or a time at which the *Aquila* is to respond. This ALARM can be set with the familiar ASCII format of commands, or, for you high speed buffs, an efficient packed decimal format.

SETA	**hh:mm:ss:ff**	Set alarm (normal ASCII format) (polled mode).
SETP	**hmsf**	Set alarm (packed decimal format) (polled mode).

Upon reception of this command, the *Aquila* will patiently sit and wait until the incoming time code is equal to or greater than the alarm time, at which time the *Aquila* will respond with one of the following:

Ahh:mm:ss:ff<CR><LF> (if alarm was set with SETA)
-or-
Phmsf<CR> (if alarm was set with SETP)

Continuous Reporting Once Per Second

If either of the EPOCH MODEs of operation is selected, the *Aquila* will send time information once every second, whenever the time code frame value equals zero. In the EPOCH - ASCII MODE, this time will be an ASCII string in the following form:

Thh:mm:ss:00<CR><LF>

Note: hh = hour value; mm = minute value; ss = second value.

The ECCS-UTCS MODE of operation allows faster communications than the standard ASCII protocol, and is designed to work with the ECCS Universal Theater Control System.

Time Code Status

Another way to determine whether the time code is active is to inquire the status of the time code with the Clock Status Inquiry. The format is as follows:

CLKS	Clock Status Inquiry (polled mode).

This will generate one of the following replies:

M28'CLOCK STOPPED'<CR>
-or-
M30'CLOCK RUNNING'<CR>

Time Offsets

The *Aquila* may be instructed to add or subtract a given offset to the time code that is being read. This offset may be temporary, or can be stored in nonvolatile memory for 'permanent' installations. The commands involved with this ability are as follows:

TOFS	**(+/-)hh:mm:ss:ff**	Offset incoming time by value of argument.
ORST		Reset offset to 0.
KOFF		Keep offset value for 'permanent' use.

Note: Only time code reading is affected by time offsets. Time code writing is never affected.

Last Input Time

When the input time code ceases, the *Aquila* saves the last input time, then clears the input time buffer to zero. To access the 'time stopped' register, use the following commands:

RRST	Reset 'time stopped' register to 00:00:00:00.
TSTP	Read 'time stopped' register.

The *Aquila* will respond with the 'time stopped' value as follows:

Thh:mm:ss:ff<CR><LF>

Setting Aquila Characteristics With Commands

The host computer can check and change the personality of the *Aquila* by means of the following commands:

MODE		Mode inquiry.
MODE	**C**	Set CONTINUOUS Mode.
MODE	**P**	Set POLLED Mode.
MODE	**E**	Set EPOCH - ASCII Mode.
MODE	**Z**	Set ECCS-UTCS Mode.
TPIN		Time code input (read) type inquiry.
TPIN	**arg**	Set time code input type. (Valid args = AVL, BCD, S10, S24, S25, SDF, S30.)
TPOT		Time code output (write) type inquiry.
TPOT	**arg**	Set time code output type. (Valid args = BCD, S10, S24, S25, SDF, S30.)
FWHL		Freewheel status inquiry.
FWHL	**N**	Disable Freewheeling (Freewheel no frames).
FWHL	**30**	Set maximum freewheel to 30 frames.
FWHL	**600**	Set maximum freewheel to 600 frames.
FWHL	**I**	Set infinite freewheel.
RGEN		Time Code regeneration/conversion status inquiry.
RGEN	**ON**	Enable Time Code regeneration/conversion.
RGEN	**OFF**	Disable Time Code regeneration/conversion.
BAUD		Communications baud rate inquiry.
BAUD	**arg**	Set communications baud rate. (Valid args = 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 76800.)
ECHO		Communications echo inquiry.
ECHO	**ON**	Enable communications echo.
ECHO	**OFF**	Disable communications echo.
RCRD		Record enable inquiry.
RCRD	**K**	Enable data record to *Keepfile*.
RCRD	**S**	Enable data record to SMPTE user bits.
RCRD	**O**	Disable data record.
PLBK		Playback enable inquiry.
PLBK	**K**	Enable data playback from *Keepfile*.
PLBK	**S**	Enable data playback from SMPTE user bits.
PLBK	**B**	Enable data playback from *Keepfile* and SMPTE user bits.
PLBK	**O**	Disable playback.
ARST		Auto Rest enable inquiry.
ARST	**ON**	Enable Auto Rest
ARST	**OFF**	Disable Auto Rest.

A command that successfully changes a setting will not elicit a response from the *Aquila*. An inquiry of a current setting will elicit one of the following responses:

Inquiry	Possible Responses
MODE	**RMODE:CONTINUOUS**
	RMODE:POLLED
	RMODE:EPOCH - ASCII
	RMODE:ECCS-UTCS
TPIN	**RCLOCK IN:BCD 10**
	RCLOCK IN:SMPTE 10
	RCLOCK IN:SMPTE 24
	RCLOCK IN:SMPTE 25
	RCLOCK IN:SMPTE DF
	RCLOCK IN:SMPTE 30
	RCLOCK IN:AVL 10
TPOT	**RCLOCK OUT:BCD 10**
	RCLOCK OUT:SMPTE 10
	RCLOCK OUT:SMPTE 24
	RCLOCK OUT:SMPTE 25
	RCLOCK OUT:SMPTE DF
	RCLOCK OUT:SMPTE 30
FWHL	**RFREEWHEEL:NO FREEWHEEL**
	RFREEWHEEL:30 FRAMES
	RFREEWHEEL:600 FRAMES
	RFREEWHEEL:INFINITE
RGEN	**RCLOCK REGEN:REGENERATION OFF**
	RCLOCK REGEN:REGENERATION ON
BAUD	**RBAUD RATE:#** (# = 300, 600, etc.)
ECHO	**RCOMM ECHO:ECHO ON**
	RCOMM ECHO:ECHO OFF
RCRD	**RRECORD:REC TO KEEPFILE**
	RRECORD:REC TO SMPTE
	RRECORD:RECORD OFF
PLBK	**RPLAYBACK:FROM KEEPFILE**
	RPLAYBACK:FROM SMPTE BITS
	RPLAYBACK:FROM SMP + KPFL
	RPLAYBACK:PLAYBACK OFF
ARST	**RAUTO REST:AUTO REST ON**
	RAUTO REST:AUTO REST OFF

LCD Display Light Commands

The operation of the LCD display light can be accomplished through the following software commands:

DSLT	**ON**	Turn LCD display light on.
DSLT	**ON**	Turn LCD display light off.
KDLT		Keep current display light status as power-on default.

A successful command will not elicit a response from the *Aquila*.

Switch Closure Commands

The operation of the built-in *Aquila* switch closures can be accomplished through the following software commands:

PLSn	Pulse Aquila switch #n for 1/2 second. (1<=n<=8).
PLSn;t	Pulse Aquila switch #n for t tenths of a second. (1<=n<=8).
XONn	Latch Aquila switch #n on. (1<=n<=8).
XOFn	Latch Aquila switch #n off. (1<=n<=8).
REST	Deactivate all relays.

Note that setting n = -1 results in the command addressing all relays. For example, XON-1 latches all relays on.

A successful command will not elicit a response from the *Aquila*.

Tape Deck Functions and Switch Closures

The *Aquila* switch closures may be attached to a tape deck's remote connector, allowing the tape deck transport to be controlled with software commands. This can be done using the previous switch closure commands, or by means of specialized commands provided to simplify this operation, as follows:

STOP	Pulse *Aquila* switch 'STOP' pattern for 1/2 second.
PLAY	Pulse *Aquila* switch 'PLAY' pattern for 1/2 second.
RECD	Pulse *Aquila* switch 'RECORD' pattern for 1/2 second.
FFWD	Pulse *Aquila* switch 'FAST FORWARD' pattern.
REWD	Pulse *Aquila* switch 'REWIND' pattern for 1/2 second.
PAUS	Pulse *Aquila* switch 'PAUSE' pattern for 1/2 second.

In the previous command listings you will note that a tape deck command will activate a 'pattern' of switch closures. This 'pattern' can be set to accommodate a variety of tape deck connection schemes. This is necessary because there is no standardization between tape deck remote control connections and capabilities. The commands to set the switch pattern for each command are as follows:

STOP	**bbbbbbbb**	Set *Aquila* switch 'STOP' pattern.
PLAY	**bbbbbbbb**	Set *Aquila* switch 'PLAY' pattern.
RECD	**bbbbbbbb**	Set *Aquila* switch 'RECORD' pattern.
FFWD	**bbbbbbbb**	Set *Aquila* switch 'FAST FORWARD' pattern.
REWD	**bbbbbbbb**	Set *Aquila* switch 'REWIND' pattern.
PAUS	**bbbbbbbb**	Set *Aquila* switch 'PAUSE' pattern.

The bbbbbbbb argument for each of the previous commands represents a pattern of 0's and 1's that the *Aquila* uses to activate the appropriate relays. The leftmost b represents switch 8, the next one switch 7, etc. Therefore, to set the STOP pattern to activate switches 1 and 3, the command would look like the following:

STOP	**00000101**	Set *Aquila* STOP pattern to activate switches 1 and 3.

Controlling Other Serial Devices

The *Aquila* can send ASCII text strings out the serial port when commanded. This capability is useful when using an Aquila Keepfile to control another device such as a laser disk or DVD player. The command to do this is as follows:

SEND *string*	Sends ASCII text *string* out the serial port at the current baud rate.

The contents of *string* can be any ASCII text character, however, any control character will terminate the string. When the *Aquila* sends the string, it will add the ASCII control characters <CR> ($0D) and <LF> ($0A) .

This command can be issued from the host computer. However, the *string* will simply echo back to the host computer.

Details on the Communications Format

Note: This information is supplied to those users who wish to write computer software to control and communicate with the Aquila. Nontechnical users may wish to skip this section.

Except where noted, command transmission to the *Aquila* is exclusively in ASCII characters. Each command sent to the *Aquila* must be terminated with an ASCII 'Carriage Return' <CR> (code $0D). All other control characters are ignored. An error in the command string will prompt a reply of 'Syntax Error'. Successful commands will not prompt a response.

Except where noted, response from the *Aquila* is also in ASCII format. To simplify host computer parsing of the message, most responses are prefaced by one letter. A test of this letter will allow your host program to branch to the appropriate service routine, if appropriate. Examples of possible responses follow:

T00:12:37:19<CR><LF>	Current time code value
M28'SYNTAX ERROR'<CR><LF>	Unsolicited message including message number.
RMODE:POLLED<CR><LF>	Reply to a setting inquiry.

Unless otherwise noted, messages from the *Aquila* will end with an ASCII 'Carriage Return' <CR> ($0D) and an ASCII 'Line Feed' <LF> ($0A).

Sequencer Operation

The *Sequencer* function allows the user to write time code synchronized programs for the *Aquila* to store and play back. These programs, called *Keepfiles,* are stored in the *Aquila's* nonvolatile memory. This means that the *Keepfile* will remain viable even if the *Aquila* is powered down and back up again. Of course, the *Keepfile* may be changed simply by uploading a new *Keepfile* from the host computer.

The *Sequencer* is designed to execute the *Keepfile* without the supervision of a host computer, making it perfect for controlling presentations in semipermanent installations. When Playback is enabled, the *Aquila* will automatically sequence through the *Keepfile,* with command execution driven by the time code.

The format of the *Keepfile* is an ASCII text file containing the list of commands and the times at which they are to be executed. Any of the *Aquila* commands that have been described are valid in the *Keepfile,* along with some of the new ones which are described as follows:

KEEP	Places *Aquila* in KEEP Mode. Subsequent commands are stored.
END	Terminates KEEP Mode and denotes end of *Keepfile.*
LIST	Lists contents of *Keepfile*.
EXTM hh:mm:ss:ff	Set Execution Time for subsequent commands.
*****	Denotes Comment line. Comments must be < 250 characters.

Keepfile Creation

Keepfile creation may be performed with the *Aquila* on-line through a terminal program. This tends to be useful for short routines only, because *Keepfile* editing is not yet available from the *Aquila.*

Keepfiles can be created with any word processor that can export text files with no formatting. Transfer to the *Aquila* can be performed through a terminal program UPLOAD (ASCII file protocol), or directly through the operating system. An example of the latter using MS-DOS follows, assuming the *Aquila* is attached to COM2, and the baud rate is 2400. At the DOS prompt, type:

A:\> MODE COM2:24,n,8,1	Set serial port parameters
A:\> COPY filename COM2:	Copy *Keepfile* to *Aquila*

Note: For best results, upload Keepfiles to the Aquila at a baud rate of 2400 or less.

Downloading the *Keepfile* From the *Aquila*

A terminal program such as *ProComm* can capture a *Keepfile* listing from the *Aquila* for editing in a word processor, if desired. Once communication is established with the *Aquila,* enable the capture file feature of your terminal program, and type 'LIST'. The *Aquila* will download the contents of the *Keepfile* into the capture file, which can then be edited, and subsequently uploaded.

Sample *Keepfile*

The following *Keepfile* is annotated for clarity. The actual file is made up of the **BOLD TEXT** in the left column only. The right column is a description of what is happening. The EXTM command is called a *time cue*, while the commands between time cues make up a *run block* for the leading time cue.

KEEP	;Initiate new *Keepfile*
*** Show Start**	;Comment
EXTM 00:00:05:00	;Set execution time for next commands
PLS1	;Pulse Relay #1 at showtime = 5 secs
PLS2	;Pulse Relay #2 at showtime = 5 secs
EXTM 00:00:10:00	;Set execution time for next commands
PLS3	;Pulse Relay #3 at showtime = 10 secs
XON4	;Latch Relay #4 on at showtime = 10 secs
END	;End of *Keepfile*

Sequencer Playback Operation

Once the *Keepfile* has been generated and stored in the *Aquila*, execution is completely automatic. With PLAYBACK ON, the contents of the *Keepfile* will be executed as the *Aquila* is either reading or generating time code. When PLAYBACK is OFF, execution of the *Keepfile* is inhibited.

One additional option available with *Keepfile* operation is that of automatic deactivation of outputs (relays or TTL bit outputs) upon the cessation of time code. This is controlled by the Auto Rest function. With AUTO REST ON, the outputs will return to their rest position in the absence of an active time code. With AUTO REST OFF, the outputs will maintain their status in the absence of time code.

Sequencer Record Operation

Using the Parallel Expander Board (PEB) described in the next section, the *Aquila* can function as a data recorder. In the record mode, ports 1 through 4 are automatically set as inputs, and can monitor the status of TTL bits or switch closures. The signals read at ports 1 through 4 are echoed at ports 5 through 8 respectively, which are automatically designated as outputs.

As time code is read (or generated), the *Aquila* monitors the signals at ports 1 through 4. Whenever any bit changes state, the *Aquila* records the time of the change, as well as the change itself. This information is stored in RAM until the cessation of time code. As soon as time code stops, the *Aquila* stores the information as a *Keepfile* in nonvolatile memory. As before, the *Keepfile* can be executed using the previously described *playback* function.

To obtain the maximum number of recorded events, the *Aquila* must have at least 32K of RAM. Most *Aquila's* delivered with the PEB option should have the correct RAM chip to support the RECORD function to its fullest. Contact ECCS for more information.

Note: This function is disabled on units without the PEB option.

The WAIT Command and Relative Timing

A command called WAIT can be used to force delays between the execution of multiple events. This can be handy when a series of related events must happen, and the timing of subsequent events depends on the execution time of the first event. The format of the command is as follows:

WAIT	**arg**	Forces a delay of (arg) seconds between events (00.00<arg<99.99).

In the following example, three relays must be pulsed at one second intervals. Only the execution of the first pulse will be triggered by the time code. The remaining pulses will occur at fixed intervals after the first pulse.

```
 KEEP                       ;Initiate new Keepfile
* WAIT COMMAND EXAMPLE      ;Comment
EXTM 00:00:05:00            ;Set execution time for pulse commands
PLS1                        ;Pulse Relay #1 at showtime = 5 secs
WAIT 1                      ;Wait 1 second
PLS2                        ;Pulse Relay #2
WAIT 1                      ;Wait 1 second
PLS3                        ;Pulse Relay #3
EXTM 00:00:06:00            ;Set execution time for next commands
XON4                        ;Latch Relay #4 on at showtime = 6 secs
END                         ;End of Keepfile
```

In this example, the pulse and wait commands make up a run block that starts at an execution time of 5 seconds and lasts until the final pulse occurs at 7 seconds. In the meantime, a second run block is initiated at 6 seconds, while the first run block is still in progress. This is legal. If fact, up to eight run blocks containing WAIT commands can be active simultaneously. However, if an attempt is made to start a ninth run block containing a WAIT, an error message will be displayed ('TOO MANY WAITS') and the run block will fail.

Loops

Repetitious events can be handled in the *Keepfile* using loops. In order to define a loop, the start point, the end point, and the number of repetitions must be defined. The commands used to define loops are as follows:

LPST	**loop#**	Defines start of loop # (1<=loop #<=8).
RPT	**loop#,times**	Defines end of loop # (1<=loop #<=8) and sets # of repetitions (0<=times<=9999).

In the following example, relay #1 must be pulsed at one second intervals. The 'on' time should be 0.1sec, while the off time should be 0.9 sec. At the same time, relays #2 and #3 should alternate evenly, with one cycle requiring 1/2 second. Both loops should start at show time of 5 seconds and last 100 seconds.

```
KEEP                        ;Initiate new Keepfile
* 2 SIMULTANEOUS LOOPS      ;Comment
EXTM 00:00:05:00            ;Set execution time for loop #1
LPST1                       ;Define start of loop #1
XON1                        ;Turn on relay #1
WAIT .1                     ;Wait 0.1 second
XOF1                        ;Turn off relay #1
WAIT .9                     ;Wait 0.9 second
RPT1,99                     ;Repeat loop #1 99 more times (total 100)
EXTM 00:00:05:00            ;Set execution time for loop #2
LPST2                       ;Define start of loop #2
XON2                        ;Turn on relay #2
XOF3                        ;Turn off relay #3
WAIT .25                    ;Wait 0.25 second
XOF2                        ;Turn off relay #2
XON3                        ;Turn on relay #3
WAIT .25                    ;Wait 0.9 second
RPT2,199                    ;Repeat loop #2 199 more times (total 200)
XOF3                        ;Turn off relay #3
END                         ;End of Keepfile
```

In this example, the loops make up two separate run blocks that start at the same execution time of 5 seconds and last for 100 seconds. The loops function simultaneously, but independently. As with WAITs, up to eight LOOPs can be active simultaneously. If an attempt is made to start a ninth LOOP, an error message will be displayed ('TOO MANY LOOPS') and the ninth LOOP will fail.

Eight looping 'frames', loop #1 through loop #8, are available for use. A particular loop frame may be used more than once in a *Keepfile,* but at any instant in time a particular loop frame can only be used in one loop. For example, if loop #1 is in progress in one run block, and is called on again in another run block before it has finished the first, an error message will be displayed ('LOOP IS BUSY') and the second instance of loop #1 will fail.

The next example appears similar to the last one. In fact, it differs in only one detail. Both loops appear in the same run block, one following the other. What will happen?

```
KEEP                        ;Initiate new Keepfile
* 2 SEQUENTIAL LOOPS        ;Comment
EXTM 00:00:05:00            ;Set execution time for loop #1
LPST1                       ;Define start of loop #1
XON1                        ;Turn on relay #1
WAIT .1                     ;Wait 0.1 second
XOF1                        ;Turn off relay #1
WAIT .9                     ;Wait 0.9 second
RPT1,99                     ;Repeat loop #1 99 more times (total 100)
LPST2                       ;Define start of loop #2
XON2                        ;Turn on relay #2
XOF3                        ;Turn off relay #3
WAIT .25                    ;Wait 0.25 second
XOF2                        ;Turn off relay #2
XON3                        ;Turn on relay #3
WAIT .25                    ;Wait 0.9 second
RPT2,199                    ;Repeat loop #2 199 more times (total 200)
XOF3                        ;Turn off relay #3
END                         ;End of Keepfile
```

In this example, the loops are in the same run block. This means that the first loop will execute until it is finished. At that time, it will 'fall through' to the second loop, which will start. In this case, the loops execute sequentially instead of simultaneously.

The 64 Bit Parallel Expansion Board (PEB)

This option adds eight 8 bit TTL compatible parallel ports to the *Aquila.* Ports five through eight are permanently configured as output ports, while ports one through four can be user configured as input or output ports. All ports can be addressed in a 'byte-wide' (8 bit) fashion by means of the following commands:

POKE ***p,d***	Enables port *p* as output port and places binary value of (decimal #*d*) at port *p.* (1 <= *p* <= 8) (0 <= *d* <= 255)
PEEK ***p***	Enables port *p* as input port, reads binary value at port *p,* and returns that value as a decimal number. (1 <= *p* <= 4)
RAMP ***p,r,v***	Enables port *p* as output port, ramps the value of the port from the current value to the new value *v*, in the ramp duration *r* seconds. (1 <= *p* <= 8) (0 <= *r* <= 99) (0 <= *v* <= 255)

As indicated above, the PEEK command prompts the *Aquila* to respond with the value present at the port in question. The format of the response is as follows:

RPEEK*p*=d	where *p* is the port number (1-4) and *d* is the value (0-255)

Alternatively, individual output bits may be commanded to latch or pulse. The commands used to activate individual bits are as follows:

XON ***p,b***	Enables port *p* as output port and activates bit *b.*
XOF ***p,b***	Enables port *p* as output port and deactivates bit *b.*
PLS ***p,b***	Enables port *p* as output port and activates bit *b* for 1/2 second.
PLS ***p,b;t***	Enables port *p* as output, activates bit *b* for t tenths of a second.

The value of -1 can be used as a wild card in the p and b arguments of the previous commands. For example, **XON p,-1** will cause all bits of port p to latch on. **XOF -1,b** will cause bits b of all ports to latch off. And, of course, **PLS -1,-1;10** will cause all 64 bits to pulse for one second.

Note that the bit commands conform to standard TTL 'active low' terminology, in that the 'OFF' state is 'HIGH' (voltage > 2.4V), and the 'ON' state is 'LOW' (voltage < 0.8V).

NOTE: Upon initialization, all ports are placed in a high impedance state, and are therefore disabled. If the Aquila is to be controlling sensitive or possibly dangerous equipment, then care should be taken to insure that all input pins on the attached equipment are 'pulled up' or 'pulled down' by means of the appropriate resistors.

These ports are available as two female DB37 connectors on the rear of the *Aquila*. The uppermost DB37 contains ports five through eight, while the lower DB37 contains ports one through four. The pinouts for these connectors are shown in Tables 7 and 8.

Table 7 - DB-37 64 Bit PIO Port A

Pin#	Signal
19	PORT 1, BIT 1
37	PORT 1, BIT 2
18	PORT 1, BIT 3
36	PORT 1, BIT 4
17	PORT 1, BIT 5
35	PORT 1, BIT 6
16	PORT 1, BIT 7
34	PORT 1, BIT 8
14	PORT 2, BIT 1
32	PORT 2, BIT 2
13	PORT 2, BIT 3
31	PORT 2, BIT 4
12	PORT 2, BIT 5
30	PORT 2, BIT 6
11	PORT 2, BIT 7
29	PORT 2, BIT 8
10	PORT 3, BIT 1
28	PORT 3, BIT 2
9	PORT 3, BIT 3
27	PORT 3, BIT 4
8	PORT 3, BIT 5
26	PORT 3, BIT 6
7	PORT 3, BIT 7
25	PORT 3, BIT 8
6	PORT 4, BIT 1
24	PORT 4, BIT 2
5	PORT 4, BIT 3
23	PORT 4, BIT 4
4	PORT 4, BIT 5
22	PORT 4, BIT 6
3	PORT 4, BIT 7
21	PORT 4, BIT 8
1	GROUND

Table 8 - DB-37 64 Bit PIO Port B

Pin#	Signal
19	PORT 5, BIT 1
37	PORT 5, BIT 2
18	PORT 5, BIT 3
36	PORT 5, BIT 4
17	PORT 5, BIT 5
35	PORT 5, BIT 6
16	PORT 5, BIT 7
34	PORT 5, BIT 8
15	PORT 6, BIT 1
33	PORT 6, BIT 2
14	PORT 6, BIT 3
32	PORT 6, BIT 4
13	PORT 6, BIT 5
31	PORT 6, BIT 6
12	PORT 6, BIT 7
30	PORT 6, BIT 8
10	PORT 7, BIT 1
28	PORT 7, BIT 2
9	PORT 7, BIT 3
27	PORT 7, BIT 4
8	PORT 7, BIT 5
26	PORT 7, BIT 6
7	PORT 7, BIT 7
25	PORT 7, BIT 8
6	PORT 8, BIT 1
24	PORT 8, BIT 2
5	PORT 8, BIT 3
23	PORT 8, BIT 4
4	PORT 8, BIT 5
22	PORT 8, BIT 6
3	PORT 8, BIT 7
21	PORT 8, BIT 8
1	GROUND

SMPTE User Bit Operation

Each SMPTE time code frame of information has 32 bits of information storage capability that can be defined by the user, hence the name *user bits*. *Aquila* units outfitted with a PEB and firmware version 1.90 or later have the ability to store and play back control bit information in these *user bits*. Each SMPTE user bit is mapped to an input bit on Ports 1-4, and an output bit on Ports 5-8. This capability can be used in conjunction with or instead of *Sequencer Keepfile* programming, explained elsewhere in this manual.

Data Storage With SMPTE Bits

The process of storing SMPTE data bits on magnetic tape is identical to the process of writing time code information to a tape, because that is exactly what is happening. The time code information is still present. The 32 user bits are additional bits that are always present, but ignored during simple time code operations.

The source of the data to be recorded may be TTL bits from another computer control system, or they may be simple switch closures. Up to 32 bits or switches may be attached to the PEB inputs bits and Ports 1-4. The *Aquila* time code out connector must be attached to the input of a tape deck channel, or another comparable storage medium. By means of the *Aquila* front panel menu, set the *Data Record* option to *Record to SMPTE.*

To begin data storage, start the tape deck in the record mode and press the 'RUN' button on the *Aquila* to start time code generation. As soon as the *Aquila* begins time code generation, it will watch the input bits on the 'PEB' ports and write their status to the user bits every SMPTE frame.

All bits are read simultaneously, at the beginning of each frame, to insure synchronicity. Any input bit activity must last at least one frame in order to be recorded.

SMPTE User Bit Data Playback

To play back the data encoded into the SMPTE user bits, attach the *Aquila* time code in connector to the output of the tape deck channel containing the information. By means of the *Aquila* front panel menu, set the *Playback* option to *Playback from SMPTE bits*. Start the tape deck in the play mode. As soon as valid SMPTE time code data has been encountered, the output bits on PEB ports 5-8 will reflect the status of the SMPTE user bits being read.

All 32 bits are written to the ports simultaneously, at the end of the time code stream, to assure synchronicity. If a bad time code frame is read, all bits will maintain their previous status until a valid frame is encountered. If valid time code ceases and the *auto rest* function is engaged, the outputs will disengage after the *freewheel* time is exceeded.

Combined Sequencer and SMPTE User Bit Data Playback

Sequencer and SMPTE data playback can be combined. As before, attach the *Aquila* time code in connector to the output of the tape deck channel containing the SMPTE user bit information. By means of the *Aquila* front panel menu, set the *Playback* option to *Playback from SMPTE bits and Keepfile*. Start the tape deck in the play mode. As soon as valid SMPTE time code data has been encountered, the output bits on PEB ports 5-8 will reflect the status of the SMPTE user bits being read.

At the same time, *Sequencer Keepfile* commands will be executed normally. If a *Sequencer Keepfile* command conflicts with the SMPTE user bit information, the *Keepfile* command will have precedence and will override the user bit information.

If any *Keepfile* command such as **POKE**, **XON**, **XOF**, or **PLS** is directed to any of the PEB output bits associated with the user bits, these PEB output bits will assume a *locked* status, and will respond only to *Keepfile* commands while ignoring the user bit information. This *locked* status will be maintained until it is released with an *unlock bit* command, **UBIT**.

In addition, the *Keepfile* command **LBIT**, or *lock bit*, can be used to *override* user bit information for just that bit without forcing the bit to any particular condition.

The *Aquila* commands that affect combined *Keepfile* and SMPTE user bit playback are as follows:

LBIT *p,b*	Lock bit *b* on port *p* and ignore SMPTE user bit information on this bit.
UBIT *p,b*	Unlock bit *b* on port *p* and resume SMPTE user bit information status.

SMPTE Bit vs. Sequencer Operation

Sequencer operation and SMPTE bit operation share the same goal, that of show control. The methods used to accomplish this goal are extremely varied, as are the strong and weak points of each method. What follows is a tabular evaluation of each method, to be used as an aid towards making the best selection for a particular application.

***Sequencer* show control**	**SMPTE bit show control**
Electronic data storage	Magnetic tape data storage
Highly reliable, stable, predictable	Subject to wear and data loss
Different shows need program reloads	No changes required for different shows
Available memory limits size of shows	Virtually unlimited show data
Any *Aquila* resource may be programmed	Limited to 32 digital bits of control

The Octal Relay Board (ORB)

Rack mount *Aquila* units with the PEB option can be outfitted with additional switch closures using the ECCS octal relay boards. Up to four Octal Relay Boards (32 additional switch closures) can be added to each rack mount unit. The switch closures are accessible on the rear of the rack unit as DB25 male connectors, with pinouts matching those of the built-in relays. Each Octal Relay Board (ORB) terminates in one of these connectors.

Each of the Octal Relay Boards are driven by one port of the PEB. Port assignment is done at the factory, starting at the highest port (8) and working down. This means that an *Aquila* with one ORB will use port 8 to control the switch closures, while an *Aquila* with three ORBs will use ports 8, 7, and 6 to control the switch closures.

The ORBs are controlled by means of the usual PEB family of commands. These include **PLS, XON, XOF,** and **POKE.**

DB25 Switch Closure Connector Pinout for the ORB

1	NC1	14	NO1
2	C1	15	NC2
3	NO2	16	C2
4	NC3	17	NO3
5	C3	18	NC4
6	NO4	19	C4
7	NC5	20	NO5
8	C5	21	NC6
9	NO6	22	C6
10	NC7	23	NO7
11	C7	24	NC8
12	NO8	25	C8
13	n/c		

Power Supply Configuration for International Operation

Most *Aquila* units are shipped from the factory configured for 120VAC/60Hz domestic United States operation. The conservative power supply design of the *Aquila* allows this configuration to operate correctly down to 100VAC/50Hz.

Aquila units using the Revision A and later motherboards can be easily adapted to function from a 220VAC power source. To reconfigure the unit to operate at the higher line voltage, follow this procedure:

1. Turn off the *Aquila*, and disconnect it from any power source.
2. Remove the two screws that attach the front panel assembly to the case.
3. Grasping the plastic bezel, gently pull the front panel assembly from the front of the case. The assembly may be allowed to dangle by the ribbon cables if care is taken not to stress the connections.
4. Remove the steel cover by pulling the cover forward and sliding it off the extruded aluminum base.
5. Examine the motherboard and locate the revision identifier. It must be a revision A or later board. *Revision 0 boards cannot be adapted*. If your unit has a Rev. 0 motherboard and you would like to operate it on 220VAC, contact ECCS about a motherboard upgrade.
6. On Rev. A and later boards, locate the 4 pin jumper labeled **W12.** It is located behind the transformer and between the fuse and the rectifier diodes. The pins are numbered 1, 2, 3, and 4 from the W12 label.
7. The configuration of this jumper determines the line voltage that the power supply expects.
   A. For 120VAC operation, pins 1 and 2 should be shorted together, and pins 3 and 4 should be shorted together.
   B. For 220VAC operation, pins 2 and 3 should be shorted together, and pins 1 and 4 should remain disconnected.
8. Replace the steel cover by sliding it on from the front of the case. Ensure that no wires get pinched or scraped as the cover is replaced.
9. Reinstall the front panel assembly carefully and replace the two screws.

The *Aquila* units are shipped with a power cord set that is compatible with the domestic United States standard. The user is responsible for supplying the appropriate cord set for his or her location.

Using the *Aquila* With the UTCS

The *Aquila* is used with the *Universal Theater Control System* by ECCS as the time code synchronizing unit. *Aquila* units shipped for use with the UTCS include a time code cable as described earlier in this manual, and an appropriate serial communication cable to be used with the *Hercules*. This provides a turn key package that includes everything necessary to synchronize your presentation.

For optimum performance, set the *Aquila* operating characteristics as follows, using the front panel menu. Consult section 2 of this manual for more information on how to use the front panel menu.

Aquila Characteristics and Settings for Use With the UTCS

Characteristic	Selection	Comment
MODE:	ECCS - UTCS	This Mode *must* be used.
CLOCK TYPE IN:	SMP30	Recommended, but type must match source.
CLOCK TYPE OUT:	SMP30	Recommended.
FREEWHEEL:	30 FRAMES	Recommended.
REGENERATION:	OFF	Recommended.
BAUD RATE:	19200	*Must* use this baud rate.
COMM ECHO:	OFF	*Must* be OFF.
RECORD DATA:	OFF	Recommended.
PLAYBACK:	------	User's choice, usually OFF.
AUTO REST:	------	User's choice, usually OFF.
LOCK:	ON	User’s choice, usually ON.

Using the *Aquila* With Pangolin Products

The *Aquila* is recommended by Pangolin Laser Software for use with the *LaserShow Designer* family of laser graphic software products to provide time code synchronization capabilities to laser show scripting.

Aquila units shipped for use with these products include a time code cable as described earlier in this manual, and an appropriate serial communication cable. This provides a turn key package that includes everything necessary to synchronize your laser show script.

For optimum performance, set the *Aquila* operating characteristics as follows, using the front panel menu. Consult section 2 of this manual for more information on how to use the front panel menu.

Aquila Characteristics and Settings for Use With The Pangolin

Characteristic	Selection	Comment
MODE:	POLLED	Polled Mode *must* be used.
CLOCK TYPE IN:	SMP30	Recommended, but type must match source.
CLOCK TYPE OUT:	SMP30	Recommended.
FREEWHEEL:	30 FRAMES	Recommended.
REGENERATION:	OFF	Recommended.
BAUD RATE:	19200	*Must* use this baud rate.
COMM ECHO:	OFF	Recommended.
RECORD DATA:	OFF	Recommended.
PLAYBACK:	------	User's choice.
AUTO REST:	------	User's choice.
LOCK:	ON	User’s choice, usually ON.

For optimum computer communications, the *Aquila* and the Pangolin computer should share the same earth ground. If this is not possible, add the ground jumper wire as indicated on the drawing in Appendix A3.

Table A - Aquila Command List

Command & Argument		Description
WRIT		Start writing time code from current time out value.
WSTP		Stop writing time code (maintains time out value).
WRST		Stop writing time code and reset time out value.
TIME		Time inquiry (polled mode).
TIME	hh:mm:ss:ff	Set time out value for writing time code.
CLKS		Clock Status Inquiry.
TSTP		'Time Input Clock Stopped' Inquiry.
RRST		Reset 'Time Input Clock Stopped' Register.
TOFS		Time Offset Inquiry.
TOFS	+hh:mm:ss:ff	Add this amount to each input time.
TOFS	-hh:mm:ss:ff	Subtract this amount from each input time.
KOFF		Keep Time Offset in nonvolatile memory.
SETA	hh:mm:ss:ff	Set alarm (polled mode).
SETP	hmsf	Set alarm (packed decimal format) (polled mode).
MODE		Mode inquiry.
MODE	C	Set CONTINUOUS Mode.
MODE	P	Set POLLED Mode.
MODE	E	Set EPOCH - ASCII Mode.
MODE	Z	Set ECCS-UTCS Mode.
TPIN		Time code input (read) type inquiry.
TPIN	arg	Set time code input type. (Valid args = AVL, BCD, S10, S24, S25, SDF, S30.)
TPOT		Time code output (write) type inquiry.
TPOT	arg	Set time code output type. (Valid args = BCD, S10, S24, S25, SDF, S30.)
FWHL		Freewheel time inquiry.
FWHL	N	Disable freewheel. (freewheel no frames).
FWHL	30	Set length of freewheel to 30 frames.
FWHL	600	Set length of freewheel to 600 frames.
FWHL	I	Set infinite freewheel.
RGEN		Time Code regeneration/conversion status inquiry.
RGEN	ON	Enable Time Code regeneration/conversion.
RGEN	OFF	Disable Time Code regeneration/conversion.
BAUD		Communications baud rate inquiry.
BAUD	arg	Set communications baud rate. (Valid args = 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 76800.)
ECHO		Communications echo inquiry.
ECHO	ON	Enable communications echo.
ECHO	OFF	Disable communications echo.
VERS		Aquila software version inquiry.
DSLT	ON	Turn on display light.
DSLT	OFF	Turn off display light.
KDLT		Keep display light status (on/off).

PLSn		Pulse Aquila switch #n for 1/2 second. (1<=n<=8).
PLSn;t		Pulse Aquila switch #n for t tenths of a second. (1<=n<=8).
XONn		Latch Aquila switch #n on. (1<=n<=8).
XOFn		Latch Aquila switch #n off. (1<=n<=8).
REST		Deactivate all relays, PEB outputs, and reset flags.
STOP		Pulse Aquila switch 'STOP' pattern for 1/2 second.
STOP	bbbbbbbb	Set Aquila switch 'STOP' pattern.
PLAY		Pulse Aquila switch 'PLAY' pattern for 1/2 second.
PLAY	bbbbbbbb	Set Aquila switch 'PLAY' pattern.
RECD		Pulse Aquila switch 'RECORD' pattern for 1/2 second.
RECD	bbbbbbbb	Set Aquila switch 'RECORD' pattern.
FFWD		Pulse Aquila switch 'FAST FORWARD' pattern.
FFWD	bbbbbbbb	Set Aquila switch 'FAST FORWARD' pattern.
REWD		Pulse Aquila switch 'REWIND' pattern for 1/2 second.
REWD	bbbbbbbb	Set Aquila switch 'REWIND' pattern.
PAUS		Pulse Aquila switch 'PAUSE' pattern for 1/2 second.
PAUS	bbbbbbbb	Set Aquila switch 'PAUSE' pattern.

The following commands are used for general show control operation:

RCRD		Data record Mode inquiry.
RCRD	K	Data record from PEB ports 1-4 to *Sequencer Keepfile*.
RCRD	S	Data record from PEB ports 1-4 to SMPTE user bits.
RCRD	O	Data Record off.
PLBK		Data playback Mode inquiry.
PLBK	K	Playback data from *Sequencer Keepfile*.
PLBK	S	Playback data from SMPTE user bits.
PLBK	B	Playback data from both *Sequencer Keepfile* and SMPTE bits.
PLBK	O	Data Playback off.
ARST		AUTO REST inquiry.
ARST	ON	AUTO REST on.
ARST	OFF	AUTO REST off.

The following commands are unique to *Sequencer* operation:

KEEP		Places Aquila in KEEP Mode. Subsequent commands are stored.
END		Terminates KEEP Mode and denotes end of KEEPFILE.
LIST		Lists contents of KEEPFILE.
EXTM	hh:mm:ss:ff	Set Execution Time for subsequent commands.
*		Denotes Comment line. Comments must be < 250 characters.
WAIT	arg	Wait arg seconds before next command. (00.00<arg<99.99).
LPST	arg	LOOP START for loop #arg. (1<arg<8).
RPT	arg1,arg2	REPEAT LOOP #arg1 for arg2 repetitions. (1<arg1<8) (1<arg2<9999).

The following commands are valid with the 64 BIT PEB option only:

POKE p,d	Enables port p as output port and places binary value of (decimal #d) at port p. (1 <= p <= 8) (0 <= d <= 255)
PEEK p	Enables port p as input port, reads binary value at port p, and returns that value as a decimal number. (1 <= p <= 4)
XON p,b	Enables port p as output port and activates bit b. (1 <= p <= 8) (0 <= b<= 8)
XOF p,b	Enables port p as output port and deactivates bit b. (1 <= p <= 8) (0<= b <= 8)
RAMP p,r,v	Enables port p as output and ramps port value from the current value to new value v, taking duration r seconds. (1 <= p <= 8) (0 <= r <= 99) (0 <= v <= 255)
PLS p,b	Enables port p as output port and activates bit b for 1/2 second.
PLS p,b;t	Enables port p as output, activates bit b for t tenths of a second.
LBIT *p*,*b*	Lock bit *b* on port *p* and ignore SMPTE user bit information on this bit.
UBIT *p*,*b*	Unlock bit *b* on port *p* and resume SMPTE user bit playback status.

Miscellaneous commands:

SEND *string*	Sends ASCII *string* out the Aquila serial port.

Table B - Aquila Messages

Message Number	Message
24	SYNTAX ERROR
28	CLOCK STOPPED
29	VERSION
30	CLOCK RUNNING

Table C - DB-15 Auxiliary Connector Pinout

1	External Sync	
2	Time Code In (+)	
3	Time Code Out (+)	
4	+5V Out	
5	PB1	'STOP'
6	PB3	'SET'
7	PB5	'PREV'
8	PB7	unassigned (V1.xx)
9	Ground	
10	Time Code In (-)	
11	Time Code Out (-)	
12	PB0	'RUN'
13	PB2	'RESET'
14	PB4	'NEXT'
15	PB6	unassigned (V1.xx)

Table D - DB-9 Communication Connector Pinout

1	TxD	Transmit Data
2	/TxD	Transmit Data (inverted)
3	/RxD	Receive Data (inverted)
4	na	(through)
5	GND	Ground
6	na	(through)
7	RxD	Receive Data
8	na	(through)
9	na	(through)

Table E - DB-25 Switch Closure Connector Pinout

1	NC1	14	NO1
2	C1	15	NC2
3	NO2	16	C2
4	NC3	17	NO3
5	C3	18	NC4
6	NO4	19	C4
7	NC5	20	NO5
8	C5	21	NC6
9	NO6	22	C6
10	NC7	23	NO7
11	C7	24	NC8
12	NO8	25	C8
13	GND		

Table F - DB-37 64 Bit PIO Port A

1	GROUND
2	n/c
3	PORT 4, BIT 7
4	PORT 4, BIT 5
5	PORT 4, BIT 3
6	PORT 4, BIT 1
7	PORT 3, BIT 7
8	PORT 3, BIT 5
9	PORT 3, BIT 3
10	PORT 3, BIT 1
11	PORT 2, BIT 7
12	PORT 2, BIT 5
13	PORT 2, BIT 3
14	PORT 2, BIT 1
15	n/c
16	PORT 1, BIT 7
17	PORT 1, BIT 5
18	PORT 1, BIT 3
19	PORT 1, BIT 1
20	n/c
21	PORT 4, BIT 8
22	PORT 4, BIT 6
23	PORT 4, BIT 4
24	PORT 4, BIT 2
25	PORT 3, BIT 8
26	PORT 3, BIT 6
27	PORT 3, BIT 4
28	PORT 3, BIT 2
29	PORT 2, BIT 8
30	PORT 2, BIT 6
31	PORT 2, BIT 4
32	PORT 2, BIT 2
33	n/c
34	PORT 1, BIT 8
35	PORT 1, BIT 6
36	PORT 1, BIT 4
37	PORT 1, BIT 2

Table G - DB-37 64 Bit PIO Port B

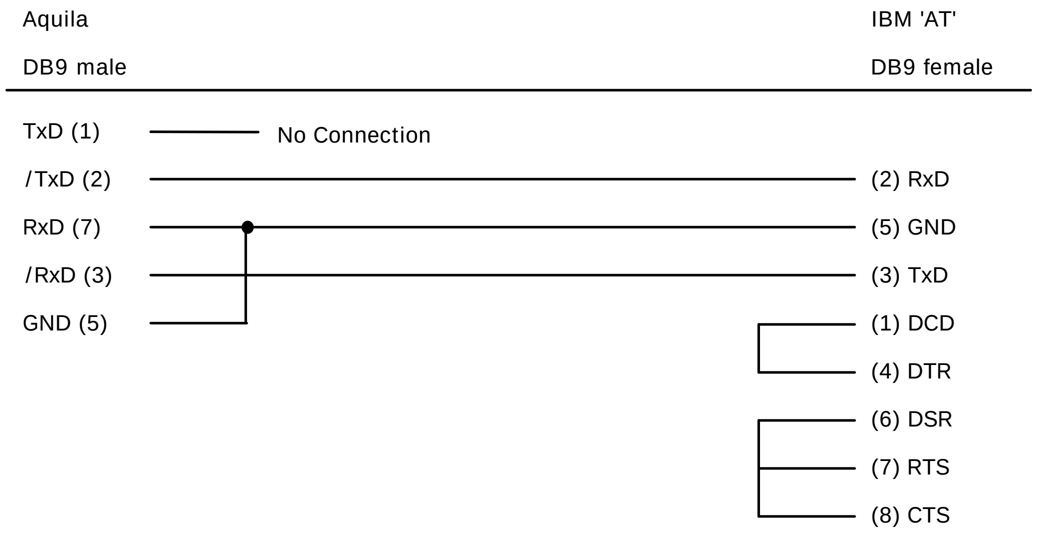
1	GROUND
2	n/c
3	PORT 8, BIT 7
4	PORT 8, BIT 5
5	PORT 8, BIT 3
6	PORT 8, BIT 1
7	PORT 7, BIT 7
8	PORT 7, BIT 5
9	PORT 7, BIT 3
10	PORT 7, BIT 1
11	n/c
12	PORT 6, BIT 7
13	PORT 6, BIT 5
14	PORT 6, BIT 3
15	PORT 6, BIT 1
16	PORT 5, BIT 7
17	PORT 5, BIT 5
18	PORT 5, BIT 3
19	PORT 5, BIT 1
20	n/c
21	PORT 8, BIT 8
22	PORT 8, BIT 6
23	PORT 8, BIT 4
24	PORT 8, BIT 2
25	PORT 7, BIT 8
26	PORT 7, BIT 6
27	PORT 7, BIT 4
28	PORT 7, BIT 2
29	n/c
30	PORT 6, BIT 8
31	PORT 6, BIT 6
32	PORT 6, BIT 4
33	PORT 6, BIT 2
34	PORT 5, BIT 8
35	PORT 5, BIT 6
36	PORT 5, BIT 4
37	PORT 5, BIT 2

Table H - DB-25 Switch Closure Connector Pinout (ORB)

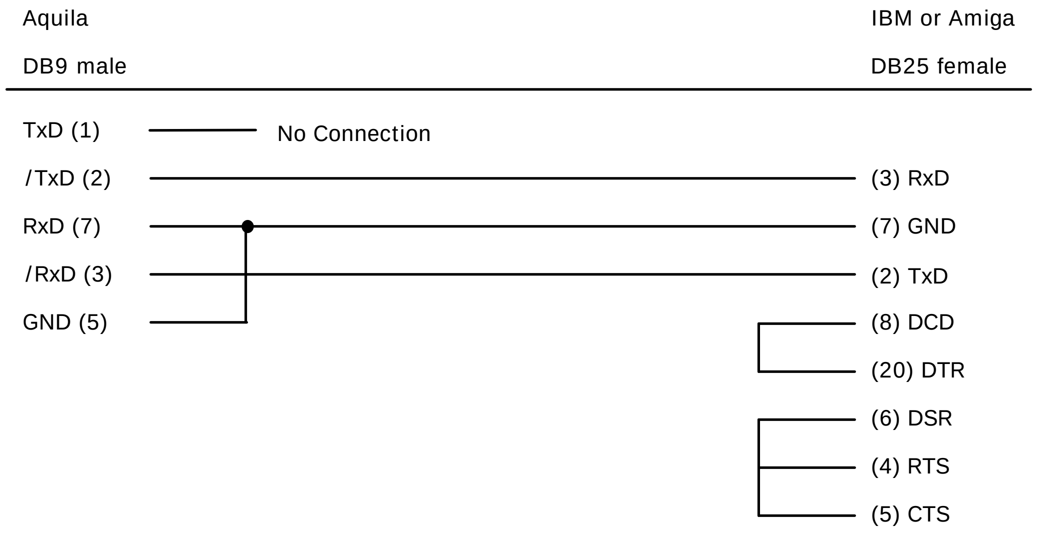
1	NC1	14	NO1
2	C1	15	NC2
3	NO2	16	C2
4	NC3	17	NO3
5	C3	18	NC4
6	NO4	19	C4
7	NC5	20	NO5
8	C5	21	NC6
9	NO6	22	C6
10	NC7	23	NO7
11	C7	24	NC8
12	NO8	25	C8
13	n/c		

Computer Communications Wiring

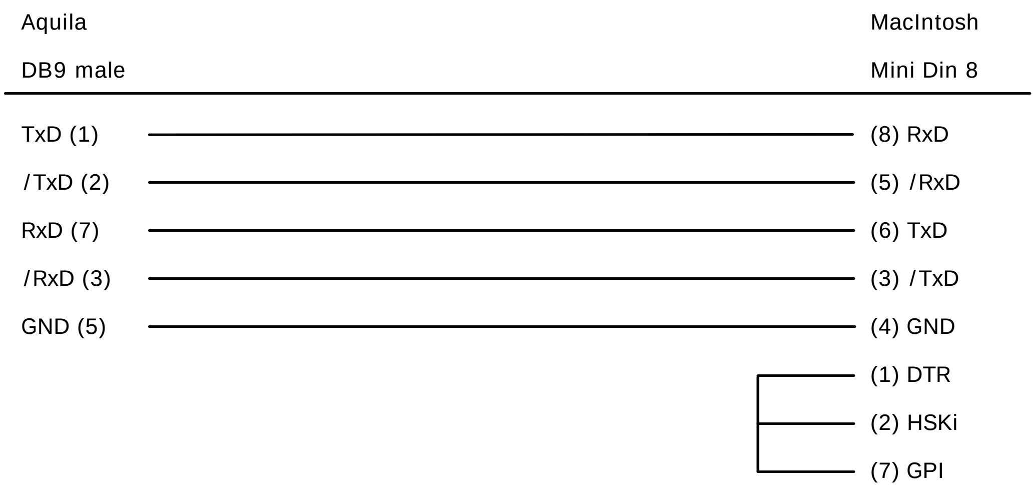
RS-232 Adapter Cable - Aquila to IBM 'AT' DB9



RS-232 Adapter Cable - Aquila to IBM or Amiga DB25



RS-422 Adapter Cable - Aquila to MacIntosh Mini Din 8


Aquila
DB9 male
MacIntosh
Mini Din 8
TxD (1)
/TxD (2)
RxD (7)
/RxD (3)
GND (5)
(8) RxD
(5) /RxD
(6) TxD
(3) /TxD
(4) GND
(1) DTR
(2) HSKi
(7) GPI