Abekas

digital disk recorder **A60**

A60

Ethernet Manual

A60 Ethernet Manual

Rev 1.3 14-AUG-88

Abekas Video Systems, Inc.

This manual describes the installation and use of the Abekas A60 Digital Disk Recorder as an Ethernet node supporting file transfer and remote login using some of the TCP/IP family of protocols. There is an application note describing some of the mechanisms behind the file transfers and the appendix lists information to allow the user to implement their own version of the file transfer routines.

Abekas Video Systems, Inc.
101 Galveston Drive
Redwood City, CA 94063
(415) 369-5111

FAX (415) 369 4777
Easylink 62796899
Telex 592712 (ABEKAS UD)
uucp Email : {allegra,decwrl,hplabs,sun,uunet}
...!pyramid!abekas!a60mail

Contents

Introduction to Ethernet and TCP/IP

Ethernet

Ethernet is a Local Area Network (LAN) Standard originally developed at Xerox Palo Alto Research Center.

Ethernet interconnects a group of computers (referred to as hosts or nodes) with a single 50 Ω coaxial cable with terminations at both ends. Data is passed serially at 10MHz in the form of packets, that is in chunks anywhere from 46 up to 1500 bytes or characters. Each packet carries addressing information to show its' source and destination.

Unlike the Public switched telephone system or a video routing matrix the single cable is shared by all the devices on the network so there are a set of rules to determine when a node can access the cable. The technique used is referred to as Carrier Sense Multiple Access with Collision Detection (CSMA-CD).

Before transmitting a node listens to confirm that nobody else is transmitting, then, as it transmits it continues to monitor the cable in case another node started transmitting at the same time. If two devices transmit simultaneously it is referred to as a collision and both devices have to stop immediately and wait a random amount of time before attempting to transmit again.

Several different manufacturers have adopted the low-level Ethernet hardware and packet specifications and built their own networks on top of it. Xerox XNS, IBM-SNA, HP-NS and DEC-DECnet are all networking systems that allow users to share resources and files and can run over Ethernet.

In the area of Personal Computers 3com Corporation and Novell are supplying File server systems based on Ethernet.

Small scale Office LAN's are mostly based on cheapernet which uses thin RG58 50 Ω cable and BNC connectors, in this case the coaxial cable is 'T'eed directly onto the Ethernet Interface in the computer. Higher level applications use better quality thick yellow cable and external transceivers which can attach to the cable with a spike-like tap.

TCP/IP

The TCP/IP protocol family is emerging as a useful common standard for network interconnection.

The strength of TCP/IP has been that it is not tied to any particular manufacturer, it is the result of extensive research since the 70's by the Advanced Research Projects Agency (ARPA) community. With backing from the DOD the emphasis for these protocols has been to interconnect different types of computers running different operating systems.

TCP/IP is now available as an add-on to most computer systems either in the form of an interface board with built in software such as the Excelan Ethernet Controllers or as an extra software package running along side a native Ethernet implementation. (The Bibliography for this manual lists some of the companies offering TCP/IP packages)

One reason for the spread of TCP/IP amongst the Computer Graphics Community has been its inclusion in the Berkeley Versions of the UNIX operating system (referred to as 4.2 BSD UNIX as opposed to the AT&T Unix System V) most graphics engines and the Workstations that control them use Unix as it is a popular operating system for software development.

TCP and IP are acronyms for 'Transmission Control Protocol' and 'Inter-network Protocol' just two of the layers in the suite of communications protocols that are required to allow transfer of data from one computer to another.

IP is the layer immediately on top of Ethernet that adds Network addressing information to the packet. These Internet addresses allow IP packets to be transferred to other networks, not just Ethernet. It is similar to the way that Containerized freight can be carried equally well by road rail or sea.

TCP provides an error free bidirectional communications channel above which other utilities, such as a remote login or file transfer, can be built.

TCP works by giving each packet a sequence number so that a message or file can be reassembled even if the packets arrive in the wrong order.

TCP also has an acknowledgement mechanism whereby the receiver replies with the latest complete sequence number it has assembled, so that should a packet get lost or delayed in the network the sender will retransmit the missing packet if it hasn't been acknowledged within a reasonable timeout period.

The third TCP mechanism is the window which limits the amount of unacknowledged data the sender can send out, so that it can't get too far ahead if the receiver is missing a packet from earlier in the message.

For the majority of File transfers or TCP connections there is no data lost. All the packets arrive in the correct order. The power of the TCP/IP protocols lies in the fact that they are not restricted to running on a single local area network.

The ARPA Internet for example combines over a 100 different networks and includes satellite links to research facilities in Europe.

When packets are passing across several networks through 'gateways' which provide an interface from one type of network to another there is more chance of a packet getting lost. There is no guarantee that all that packets will take the same route to the destination, this is possible since each packet carries separate addressing information. It is up to the gateways to decide which is the most efficient route. If during the life of the connection one of the intervening gateways or networks goes down the TCP/IP protocol is robust enough to be able to replace any unacknowledged lost data by retransmission (assuming an alternative route can be found).

Telnet and FTP

The A60 supports file transfer and remote control built on top of the guaranteed delivery TCP connections. Remote control is achieved by allowing the remote user to 'login' as if the A60 were another computer and type commands interactively. There are two alternatives for each. Firstly the official ARPA file transfer and remote login utilities called FTP (File Transfer Protocol) and Telnet. These are specifically intended to work between different Computer architectures and Operating Systems.

Unix 'r' Commands

The second alternative is the native Unix utilities 'rcp' (remote copy) 'rlogin' (remote login) and 'rsh' (remote shell). These will be popular with Unix users since they offer a less verbose user interface - file transfers are achieved by cryptic one line commands rather than FTP which normally produces a secondary prompt and requires at least three commands to transfer one file.

The Abekas A60

The A60 can be viewed as a Video Server - permitting all the rendering engines and computers in a graphics lab to share the ability to try out animation sequences and lay off rendered images without the preroll and lineup problems associated with single frame VTRs. It is a powerful shareable resource that treats all the frames or fields on a disk as separate files that can be copied to or from the A60.

The remote control provided across Ethernet is intended to be human readable in that the commands for playing or setting up segments are executed by typing
"PLAY" or
"DEFSEG 0.20 1.30"
a user sitting at a workstation can easily preview frames or animations without the need for a multiple control panels.

As an Ethernet device, the A60 appears to be just another node that files can be transferred to in the same way they would be moved from one computer system to another, there is no need for a separate VTR controller or special software.

Installation on Ethernet is just a question of the plugging the A60 into a Transceiver, which is a small box that provides an isolated interface to the Ethernet Coaxial Cable. The System Manager then specifies an Internet address for the machine which has to be entered on the A60 control panel. The other computers on the network can then use this Internet address to access the services supported by the A60.

TCP/IP Application Notes

This section gives a brief outline of the functions of each of the protocols used in the A60 and the way they interact.

Typical File Transfer

Take for example an rcp transfer on a Unix system. To invoke rcp the user types :

```
unix% rcp file.yuv a60:37
```

which causes a connection to be opened to the rcp server on the Abekas A60. The hostname is the name used to refer to the A60 on the users' machine. Somewhere there will be a file (/etc/hosts on unix) which gives the mapping between the name (or some other optional alias) and the Internet Address that has been assigned for the A60.

A typical entry in /etc/hosts has the following form :

```
192.5.200.9          a60
```

Rcp will first find the Internet address of the remote host by referring to the /etc/hosts file. Then it has to find the lower level Ethernet address that corresponds to the machine with that Internet address. The Ethernet address will either be the address of the A60 itself or a Gateway device on the local ethernet through which the A60 can be reached.

In some cases the host computer may have retained this information from a previous transaction, but for the first transfer to an unknown remote host the local host has to resolve the Internet - Ethernet address mapping. To do this it uses the Address Resolution Protocol (ARP).

ARP involves sending a broadcast packet to all the hosts on the network.

Only the host with the required Internet address, or a Gateway that can reach it, will reply with a ARP reply packet supplying the requested Ethernet address.

Once the local host knows how to reach the A60 over Ethernet it can open up a TCP connection to the rshell port on the A60.

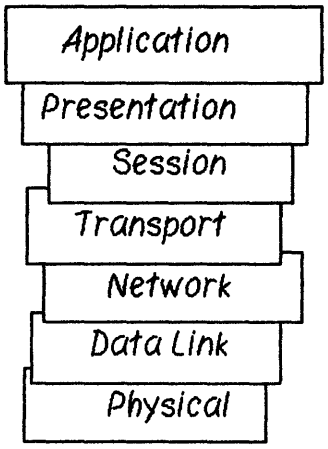
The A60 which has been passively waiting for a connection to be made to its' rshell port responds with an opening message and the rshell program passes the users' name and the destination filename to the A60.

Since the A60 does not support username validation, passwords or accounting the transfer will proceed immediately with the transfer. The A60 will seek to frame 37 and the data will be written to the framestore and displayed as it arrives. When the transfer is complete the A60 will record the frame and close the connection.

Layered Model

The transfer outlined above used several program modules, one was responsible for transmitting and receiving packets, another assembled the incoming data into order and retranmitted unacknowleged outgoing packets. At a higher level there was a seperate module that dealt with converting data formats. Each of these modules has a well defined interface to the modules below wether it is passing a packet to the Ethernet module or reading and writing data to or from a TCP connection.

The ISO standards authority has proposed an Open Systems Interconnection model consisting of seven layers. The reason for dividing the Protocols into layers is so that different protocols at the same level can be interchanged to provide the same function for the layers above.



OSI diagram

The TCP/IP protocols occupy the Transport and Network layers and can be applied to several lower level network architectures. In this case the two lower levels - that is the Physical layer which defines the connectors and signal voltages, and the Data link layer which defines the way data is passed between two pieces of equipment on the same local network, are both defined by the IEEE 802.3 standard (which in turn is based on the Xerox Ethernet V2). Another comparable Network standard is X.25 which is used for public packet switched services such as Tymnet.

Physical layer

Ethernet was developed at Xerox PARC and is based on the concept of a baseband Carrier Sense Multiple Access with Collision Detection. (CSMA-CD)

Transceivers

The host computer will be coupled onto the Ethernet Coax by a device called a transceiver. In some instances (particularly in the case of 'Cheapernet') the transceiver can be part of the Ethernet interface board in the host computer. The Remote transceivers can be up to 50m away from the computer and typically come with a plug in module to allow either a spike tap, N-series or BNC connectors to interface to the cable.

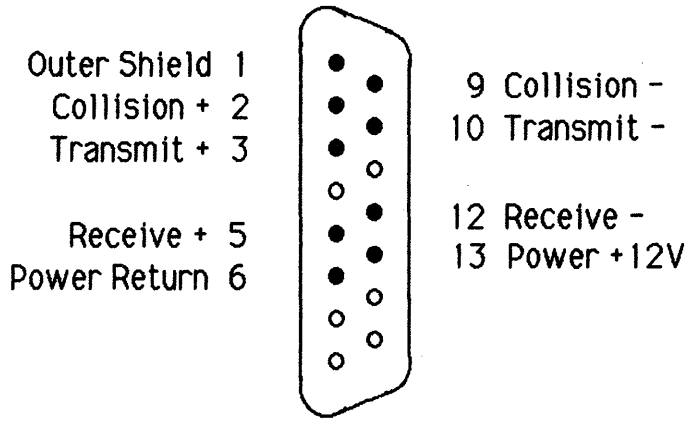
Cheapernet uses thin RG58 50-Ω coax and BNC style connectors rather than the high grade 10mm thick Yellow coax which permits the non-intrusive "vampire tap" transceivers to be spiked into it and removed without having to take the network down.

The thick coax can be used for networks up 100 nodes on 500m of cable whereas Cheapernet is limited to 30 nodes on 185m of cable. Only two repeaters are allowed on a local network because of the propagation delays through them. The transceivers should be placed at multiples of 2.5metres on the cable.

Different Ethernet Standards

There are three different Ethernet standards : Ethernet V1, V2 and the Newer IEEE 802.3 standard.

The are electrical differences between the different standards mainly concerning the grounding requirements for the transceiver. For Ethernet V1 and V2 the connector shell, cable shield and pin one of the connector are all connected together whereas for IEEE 802.3 the cable shield is separate from pin 1 ground. The grounding should be achieved at the host end of the transceiver cable.



Transceiver (AUI) cable pinout

SQE

Some transceivers support SQE or Signal Quality Error (sometimes called heartbeat) in which case they will return a collision in the gap at the end of every transmission from the host. This provides a way of assuring that the collision detection circuitry in the host interface is functioning correctly. Ethernet V1 does not provide for SQE.

Data Link Layer

The Data Link layer provides for source and destination addresses and a trailing CRC allows the integrity of the whole packet to be checked. Ethernet and IEEE specifications differ in a few minor respects, The IEEE spec states that the field immediately following the source address is an optional length, however the Ethernet receiving hardware is capable of determining the packet length by other means. The original Xerox Ethernet spec defines this as a type field which is used to resolve the next protocol above in the hierarchy. The types used for the ARPA family of protocols are intentionally chosen to be illegal lengths (eg larger than the maximum permissible) so as to remove any confusion between the IEEE and Xerox implementation of Ethernet.

Ethernet Addresses

Ethernet addresses are six bytes conventionally written as six hex numbers separated by colons. Ethernet addresses are intended to be unique for any piece of equipment that conforms to the IEEE 802.3 standard.

Abekas Equipment will be in the range 00:00:76:XX:XX:XX

Using the Ethernet Broadcast packet mechanism it is possible for the local host to send a packet to all the devices on the local network rather than to one specific address. This address is FF:FF:FF:FF:FF:FF .

Network Layer

The Network Layer provides a packet delivery system between two hosts.

IP

The Internet Protocol adds to a packet the information required to pass it from one host to another across different networks. The intention is that a gateway that is passing the packet from one network to another need only examine the IP header to determine the routing information.

The Internet Protocol also provides a mechanism for breaking packets into smaller fragments for passing them over a network with restricted packet size.

Internet Addresses

Internet address is a four byte number conventionally expressed as four decimal numbers (0..255) separated by dots.

eg 192.9.200.5

The Internet Address can be subdivided into two parts. The first is referred to as the Network number and the second as the local address. All the machines on the same local network should have the same Network Number but different Local Addresses. If a host is asked to send a packet to a remote host with a different Network number it will assume that the remote host is on a different network and attempt to find a gateway to it.

Network Numbers for Internet sites are assigned by the ARPA authorities, the local addresses are chosen by the local system administrator. There are many Ethernet networks that are not actually part of the Internet and only use a default Network number.

There are three classes of internet address based on the the Network number.
For a class A address the first byte of the internet address will be less than 128, the local address is then formed by the lower three bytes allowing for 4096 separate hosts on the one network.

A class B address will have a first byte in the range 128 to 191 and in this case the lower two bytes form the local address.

Class C addresses will have first byte greater than 191, only the last byte identifies the individual machine or internet node.

Internet addresses are intended to be more wide ranging than the native addressing scheme for the the local network, it is possible to connect to an Internet host across several different types of network eg Local Area Ethernet, long haul X.25 packet switched services and Local area Token ring networks.

ICMP

Internet Control Message Protocol is almost an integral part of IP. It is intended to handle error reporting from Gateways to hosts or hosts to hosts. It gives the originating host more information about the reason a packet has been rejected or cannot be delivered.

ICMP also provides an echo system used by 'ping' programs to help isolate inter-network problems.

A Selection of the ICMP Messages :

Echo Request
Echo reply
Redirect : Use alternative route
Time Exceeded : Packet died of old age
Parameter problem : Something wrong with the IP Header
Destination Unreachable, either:
Network
Host
Protocol
Port

GGP

The Gateway to Gateway Protocol Handles communication between Gateways for control purposes. It allows them to exchange routing information and keep up to date on the availability of neighboring Gateways.

ARP

The Address Resolution Protocol uses the Ethernet Broadcast Mechanism to allow a host to resolve Internet to Ethernet address mappings by asking all the hosts on the local network if any of them claim to be the required Internet address.

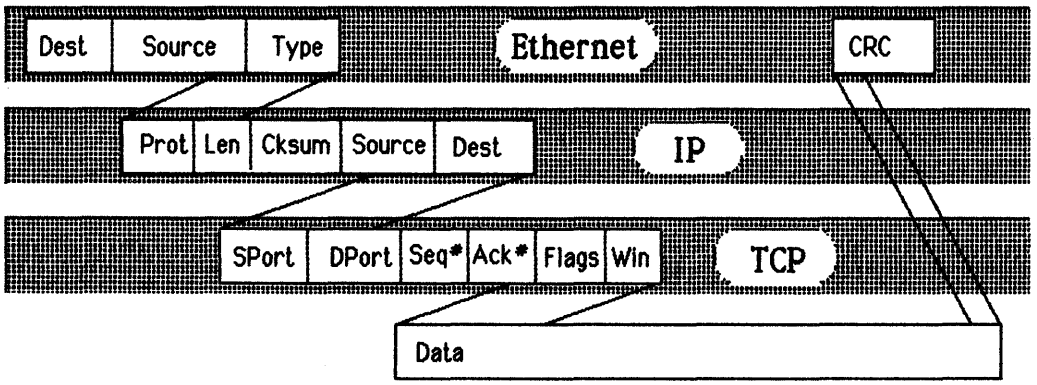
If the hosts support ARP they will decode the packet, it contains both the Internet and Ethernet Addresses of the local host and the Internet address of the host it is trying to reach. Only the requested host or a Gateway that can reach it on another network reply directly to the requesting machine.

RARP

The Reverse Address Resolution Protocol allows an Ethernet host to determine its own Internet Address by broadcasting a request to the net.

Transport Layer

The fourth OSI layer is the Transport Layer, which is concerned with creating and maintaining logical connections between individual processes on different hosts.



A Typical Data Packet

TCP

The transport layer adds a source and destination port number to the packet addresses. This allows the packet traffic to be routed to several different processes or users within a particular machine. A TCP connection is characterized by the combination of local and remote Internet addresses and the local and remote port numbers. So for instance several people may be logged in on a machine through the port assigned to the remote login service. If two of the users originate from the same machine they will have to be on different ports on their local machine. The TCP software can then identify uniquely which connection a packet belongs to.

Some of the lower port numbers (normally less than 1024) are reserved for system functions, these are *well known* port numbers that are published for other computers wishing to use a particular service. Opening connections to these ports is normally a privileged operating system function.

FTP	21
Telnet	23
rsh	514
rlogin	513

Commonly used TCP port numbers

The TCP packet header also carries Sequence and Acknowledge numbers. The Sequence number represents the position of the first byte of this packet in the transmitted data stream. The Acknowledge represents the byte after the last fully reassembled byte of the data stream received. In other words the Receiving TCP can buffer several packets that may be out of sequence but the Acknowledge number will only increase once the incoming data is complete up to that byte number.

Sequence numbers are fixed when the connection is opened, packets are exchanged with the SYN flag set to indicate the initial sequence number.

The TCP header also includes a window to indicate to the other end how much buffer space is currently available. This acts as a method of flow control since the transmitter should not continue if more than a windows' worth of data remains unacknowledged.

The transmitting side of TCP is responsible for resending a data if it is not acknowledged within a timeout period.

There is a TCP Option that allows the maximum number of data bytes in a packet to be specified, the default is 512 but some systems accept up to 1460 bytes, this leaves 40 bytes for the TCP/IP headers in the maximum sized ethernet packet of 1500 bytes.

TCP connections can be opened either Actively or Passively. In the Active case the remote port and host address are fully specified, whereas a Passive open will only specify the port number at the local end and wait for an incoming attempt to connect.

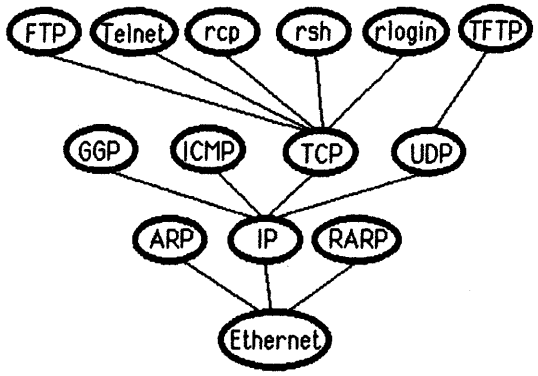
On closing the connection it is necessary for both sides to exchange and acknowledge packets with the FIN flag set.

UDP

The User Datagram Protocol is a simple extension to IP which adds only a source and destination port number and a checksum. The Datagram is a standalone packet with neither guaranteed delivery nor special sequence.
TFTP and Sun Microsystems' RPC and NFS are among the protocols built on top of UDP.

Upper Levels

For the TCP/IP protocols the top three OSI reference layers tend to be merged into one program at the highest Application layer. The OSI model also provides for a Session layer which handles user validation and mapping host names to network addresses, and a Presentation Layer which handles machine differences like byte swapping and terminal standardization.



TCP/IP Family Tree

Telnet

Telnet is a remote login program based on the concept of a Virtual Terminal. The Virtual Terminal has a set of default conditions that can be changed by *negotiation* and mutual agreement between the local and remote host. For instance echoing is done locally by default and the local host is expected to buffer lines of text until <Return> is pressed.

The Telnet connection has an escape mechanism where hex FF is the escape character. An FF occurring in the data stream is transmitted as FF FF. The escape character precedes an option negotiation which contains a code to indicate WILL, WON'T, DO or DON'T and the particular option code. Options include the ability to turn off echo and set various terminal attributes such as line length. Before an option will be implemented on both sides they have to positively agree to do it. The FF escape sequence is also used to implement 'out of band' signals such as Abort Output, Interrupt Process and Erase.

FTP

The File Transfer Protocol uses a Telnet connection for User authentication and control. The control is achieved using a command and Response Dialog which may or may not be visible to the user. Commands are of the form "USER Simon" and "STOR pic.rgb". FTP responses are preceded by a three digit code which allows a machine to assess the required action.

Some of the meanings are listed below :

1xx is a Positive preliminary reply
2xx is a Positive completion reply
5xx is a Permanent negative completion reply

x0x is a Syntax error
x2x refers to a connection
x5x refers to the file system

Example FTP transfer

In the following example each line shows a step in the transfer, the arrow indicates the direction of the message (➡ is from the host to the A60) and the carriage return and line feed characters are shown in the 'C' language notation \r and \n.

[open TCP connection to port 21 on the A60]
⬅ 220 Abekas A60 FTP (a60)
➡ USER simon\r\n
⬅ 230 User OK
➡ PORT 192,9,200,1,30,244\r\n
⬅ 200 PORT spec accepted : host 192.9.200.1 port 7924
➡ STOR 407.rgb\r\n
[Active open from A60 end port 20 to 192.9.200.1 port 7924]
⬅ 150 OK here goes
[Data Transferred] far end closes data connection when done
⬅ 226 File Transfer OK
➡ QUIT\r\n
⬅ 221 Closing control connection

TFTP

The Trivial File Transfer Protocol is built on top of UDP and provides an easy-to-implement file transfer. Data is transferred in 512 byte blocks each data packet carrying a block number. Each block has to be acknowledged before the next one can be transmitted. TFTP is used for booting diskless hosts on a local network and exchanging mail.

rcp

As is the case with most Unix applications the implementation of 'r'copy is elegantly minimal. Most of the handshaking is achieved by the transmission of a single null byte. Most of the 'r' commands use a single TCP connection although there is provision for a 'standard error' connection. In the following example "\0" indicates that a null (zero) byte is sent.

rcp pic400.rgb a60:312.rgb

➡ \0	**(no standard error)**
➡ simon\0simon\0	**(local and remote username)**
⬅ \0	**(confirm user info validated)**
➡ rcp -t 312.rgb\0	**(the command)**
⬅ \0	**(command OK)**
➡ C0666 1049760 pic400.rgb\n	**(Access flags, length, original name)**
⬅ \0	
➡ [1049760 bytes of data]	
➡ \0	
⬅ \0	

These transfers are listed in more detail in the appendix.

File Transfer

General Information

RGB Conversion

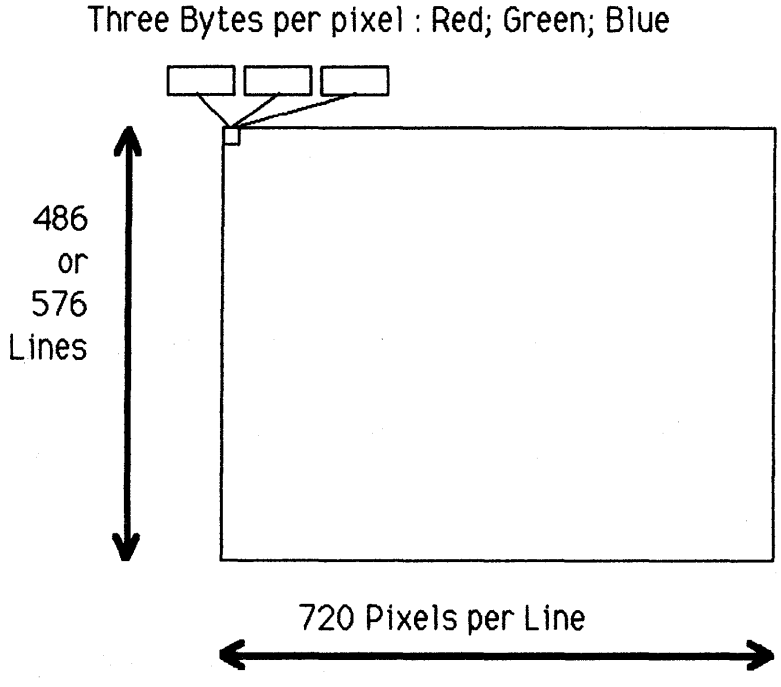
The A60 has the capability to convert raw RGB files to YUV and back again. The arithmetic is done 32 bit fixed point using the on board Multiplier Accumulator Chip. Reciprocal Anti Aliasing and Interpolation filters are used to minimize generation loss. Some slight degradation will be visible on the first pass into and out of the machine. Once the image has been bandwidth limited to YUV space it does not degrade further on successive passes. The conversion process does slow up the transfer of data however.

Transfer Times

On a lightly loaded Ethernet with a host that supports 1024 byte TCP packets a 525 line YUV image should take about 7 seconds to transfer in either direction. RGB images take approximately 25 seconds to transfer to the A60 and 33 seconds to read from the A60.

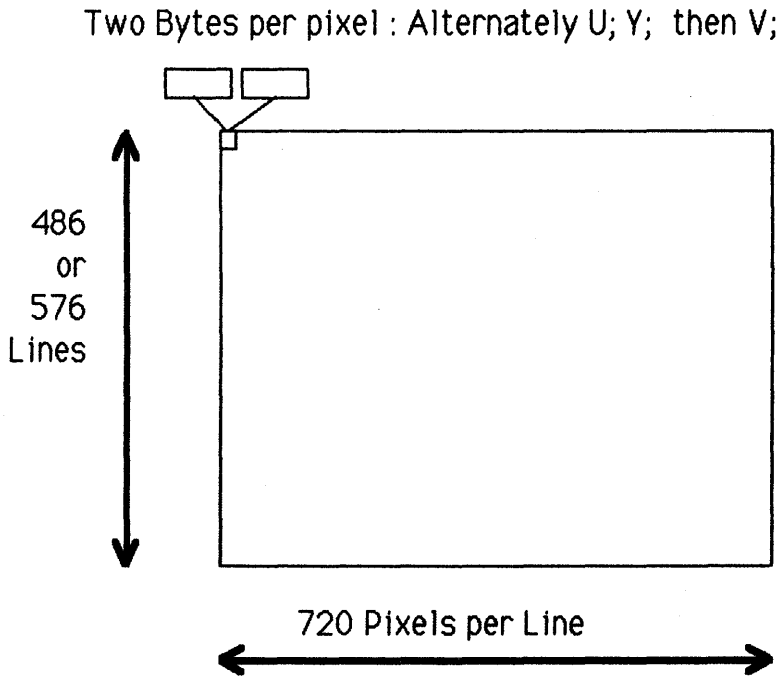
Data Format

Video data in the files transferred to and from the A60 is raw RGB or YUV data with no compression, Run Length Coding or other structure. There are 720 pixels per line and for a 525 line system there are 486 lines per frame (576 for 625 line systems).
The pixels are ordered in the same direction as the TV raster is scanned, eg left to right and from the top down.



RGB Files

Refer to the A60/A64 Digital Video Interface Manual for more information on the format of the data and the size of the frame buffer.



YUV Files

Note that the A60 does not check the length of the file transferred but files longer than a frame in length can cause the Frame buffer addressing to wrap round and cause unpredictable effects.

A60 File Name Conventions

The A60 expects a filename to contain the frame number and an optional extension .yuv **and** .rgb **are currently supported, yuv is the default. The frame number can either be given as an absolute frame number (in decimal) or a time code.**

The A60 starts at the end of the filename and works back so you can include all the directory paths you want, they are all ignored.

rcp thing400 a60:/users/simon/piccys/thing400.yuv

parses OK as 400.yuv

Be careful that filenames don't have an 'F' before the frame number since this has the special significance explained below.

Fields and Frames

Video can be transferred as Frames or Fields, A frame is twice as long as a field and the lines are interleaved in the same way as they appear on the display. A field transfer is indicated by an 'F' immediately preceding the frame number. The first field of a frame will be implied by default. To access to the second field the frame number must have a '+' appended to it. The frame number can also be specified in timecode, in which case seconds and frames are separated by dots for the first field and colons for the second.

rcp mypic.yuv a60:590

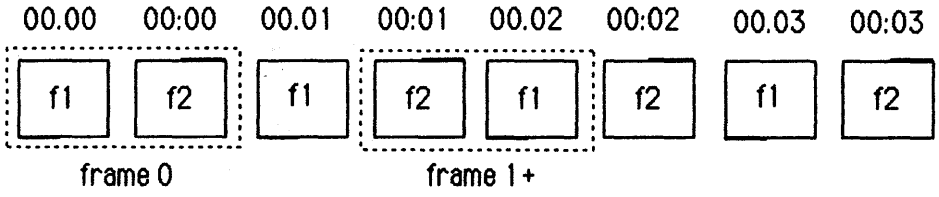
places the frame mypic at 590

rcp myfield.yuv a60:f590+ or **rcp mypic.yuv a60:f18:25**

records the file as the second field of frame 590 (18:25 in timecode on 525 line systems)

Field Dominance

The diagram below show the difference between a frame that starts with a field 1 and a field 2. The example is for 525 line systems since in 625 the upper field (eg the one with the half-line at the top) is actually field 1.



If rendered frames are being transferred to an A60 that will be used later in an editing environment that uses field two dominance (eg edits are made between field one and field two rather than on frame boundaries. It may be more appropriate to transfer the frames starting with the second field.

rcp myframe a60:20+

places the frame at fields 00:20 and 00.21 rather than 00.20 and 00:20

Write Protection

The A60 has the ability to Write Protect or Record Lockout segments of the Disk. An attempt to transfer to a Record-Locked-out section will result in an appropriate "Write Protected" error message. Note that Record Lockout can be changed using the protect and unprotect commands from the Ethernet remote software.

FTP

The following is the example of an FTP transfer files to and from the A60. Boldface type indicates what is entered by the user.

```
host% ftp a60
Connected to a60.
220 Abekas A60 FTP (a60)
230 User OK
ftp> type image
200 TYPE OK
ftp> send piccy 25
200 PORT spec accepted : host 192.9.200.6 port 1061
150 OK here goes
226 File Transfer OK
699840 bytes sent in 4.92 seconds (1.4e+02 Kbytes/s)
ftp> send sym/symb576.rgb 25.rgb
200 PORT spec accepted : host 192.9.200.6 port 1062
150 OK here goes
226 File Transfer OK
1244160 bytes sent in 29.40 seconds (41 Kbytes/s)
ftp> quit
221 Closing control connection
```

Note that the FTP MPUT command is useful for transferring a whole set of files with similar names. The file name on the host computer has to contain a frame number.

```
host% ls pic*
pic000.yuv      pic001.yuv              pic002.yuv
host% ftp -i a60
Connected to a60.
220 Abekas A60 FTP (a60)
230 User OK
ftp> type image
200 TYPE OK
ftp> mput pic*
200 PORT spec accepted : host 192.9.200.6 port 1061
150 OK here goes
226 File Transfer OK
699840 bytes sent in 4.92 seconds (1.4e+02 Kbytes/s)
200 PORT spec accepted : host 192.9.200.6 port 1062
150 OK here goes
226 File Transfer OK
699840 bytes sent in 4.92 seconds (1.4e+02 Kbytes/s)
200 PORT spec accepted : host 192.9.200.6 port 1063
150 OK here goes
226 File Transfer OK
699840 bytes sent in 4.92 seconds (1.4e+02 Kbytes/s)
ftp> quit
```

rcp

```
host% rcp piccy a60:23
host% rcp sym/symb576.rgb a60:rgb
rcp: RGB Not a valid frame number
host% rcp sym/symb576.rgb a60:24.rgb
host% rcp a60:24.rgb zzz
```

Note that rcp from A60 to '.'
(eg the same name in the current dir) is supported

```
host% rcp a60:24.rgb .
host% ls *.rgb
024.rgb
```

The A60 will generate a file in the form [F] 001 [+] .yuv. The reverse case is not supported, the A60 insists on having the filename specified explicitly.

```
host% rcp symb576.rgb a60:
rcp: . Not a valid frame number
```

By the same token directory copies or wildcards aren't accepted

```
host% rcp *.rgb a60:
rcp: A60 only supports single file transfers
```

A third party copy from another host to the A60 *does* work

```
host% rcp otherhost:frame.yuv a60:45
```

it is implemented simply by invoking rcp on the otherhost via an rsh, the reverse is not supported however since the rcp command is not supported by the A60 'shell'.

```
host% rcp a60:22.rgb otherhost:frame.rgb
rcp: A60 cannot originate transfer - no hostname table
```

Unix Script Hints

Here are some suggestions for ways to use Unix Command Files to shuffle files onto or off the A60. They all run under the 'C'shell.

```
# copy dir SMC Siggraph 87
# copies all the frames with same root name eg 'pic' in pic001.yuv
# takes 3 params : copydir rootname host baseframe
# note that frame numbers in filenames should have leading zeros
# - or ls screws up the ordering
set frame=$3
set list=`ls $1*`
foreach i ($list)
   rcp $i $2:$frame
   @ frame++
   end

# copy frames SMC NAB-88
# takes two args the base frame number and the total number to transfer
while($count > 0)
   rcp piccy.$frame $1:$frame
   @ frame++
   @ count--
   end
```

A60 Remote Control over Ethernet

The following notes outline the commands available over the A60's TCP/IP Ethernet connection. It assumes some knowledge of the operation of the A60. Refer to the A60 Operators Manual for more information on how the machine can be divided into segments and the limits on play speed etc.

Remote commands can be invoked in three ways Telnet, Rsh and Rlogin. The following examples assume the commands are given on a UNIX machine with the prompt host%

The A60 has several operating modes particularly Segment, Normal play and Loop Mode, most of the commands given here will work with the machine in any state however they can leave the machine in a different state.

Telnet

From Unix type

```
host% telnet a60
Abekas A60 Telnet
a60> play
a60> quit
```

rsh/rlogin

rsh allows a single command to be executed on the remote machine for instance

```
host% rsh a60 play -0.5
```

Both Rsh and Rlogin offer an interactive login, either

```
host% rsh a60
```

or

```
host% rlogin a60
Abekas A60 Remote Login
a60% play
a60% quit
```

the login can also be terminated by ^D or ^C

Note that it is also possible to redirect a file to the remote shell (even though this is not strictly legal Unix Syntax). To achieve the same effect between two Unix machines you actually have to type "rsh rhost csh < script"

```
host% cat script
clearseg
defseg 100 200
defseg 300 400
loopseg 1 2
host% rsh a60 < script
```

This provides an easy way to keep track of the segment list from the host computer since it is not possible to read back the segment list from the A60.

Feedback is limited to the 'where' command but this should be enough to indicate that the A60 has reached the end of a segment with a pause in it.

```
host% cat script
clearseg
defseg 100 200 pause
defseg 300 400 pause
loopseg 1 2
host% rsh a60 < script
host% rlogin a60
a60% w
165
a60% w
174
a60% w
190
a60% w
199
a60% w
199
a60% plays
a60% w
307
```

Note that out point for a segment is not inclusive eg defseg 100 200 plays frames 100 to 199.

Note the following case where information can be interpreted by a shell script

```
# Record relative
# Record a single frame of input video at the current location
set frame=`rsh a60 where`
rsh a60 record input $frame
```

General Command Information

Upper or lower case allowed
Shorter forms of commands given here can be given
Parameters separated by spaces
Only one command per line
Note that segments are numbered from one
Parameters listed in [] brackets are optional

Be warned that in the case of ambiguity the earlier command in this list is executed.

Parameters

speed
[-][0-9].[0-9] limits +/- 30 for 525 line systems (25 for 625) resolves three decimal places eg -1 1.5 0.5 .5 -30 7.125 are all legal speeds

frame
[f][+-][0-99][.:][0-99][+-] accepts absolute frame numbers with trailing '+' to indicate the second field or timecode with '.' or ':' for field 1 or 2 field mode or auto frame is not changed

eg 1 3:01 1.00.24 f231+ are all valid

seg
[1-100] segments can be numbered from 1 - 100 The segment number follows the order in which the segments were defined.

source
any of the following is permissible as a record source, it should only be necessary to type the initial letter of each option.

bars
pattern <pattern_num>
input - Digital Video in #1
aux - Digital Video in #2
<frame> - a frame number

pattern_num
(from 0) 100% Bars, 95% Bars, 75% Bars, Lin Ramp, Mod Ramp, 10 Step, X Hatch, 2T pulse, Pluge, Multiburst, Bow Tie, Digital test, 100% Combo, 95% Combo, 75% Combo

display_mode
field or frame

Commands

play [speed]
defaults to 1.0
play from the current position

stop
stop loop or play

goto [frame]
defaults to 0
goto the given field or frame

jog [offset]
defaults to +1
relative goto,
field or frame offset is determined by the current display mode

loop <in> <out> [speed]
loop play the specified segment

defseg <in> <out> [speed] [pause]
define the next segment from in to out with optional speed
speed defaults to 1.0
pause can be typed in full or just 'p'

goseg [seg]
defaults to 1
goto the 'in' point of the given segment

playseg [speed]
play thru the segment list from the current position
or alternatively resume playing after a pause

loopseg <in> <out> [speed]
loop play the specified segments

clearseg
clear the segment list

macro <macro_number>
execute the given macro

quit
close down the connection

hostname
set the hostname returned in the prompts (default 'a60')
makes it easier to use two machines

where
where returns the current frame

record <source> <in> [<out>]
records from various sources pattern, input or specific <frame> number.
Defaults to a single frame.
Using the trailing plus sign it is possible to record odd numbers of fields.
record 10 10+ will record a single field and record 11+ 13 will record three fields starting on an odd boundary.

protect <in> [<out>]
write protects the given range, the range is inclusive
in frame mode the range is rounded up to the second field

unprotect <in> [<out>]
unprotects the given range

pattern [pattern_num]
draws the given pattern into the store

mode [display_mode]
with no parameters returns the current mode.
Note that to change the mode frame or field has to be typed in full

enable
unlock the A60 Keyboard (see Disable)

disable
locks the A60 keyboard - prevents accidental use of the trackerball
can be overridden by pressing a **<number>** followed by the **<stop>** button on the keyboard.

freeze
freezes the framestore

unfreeze
unfreezes the framestore, note that unfreezing will not change the display unless the disk is already playing

input [601 input]
selects video input 1 or 2 (defaults to input 1)

recrel [n_fields]
record relative (ie like the control panel) from current position for n_fields (defaults to a frame)

man

help [command or parameter]

the help command provides information on individual commands or parameters. With no arguments it lists all the commands available. 'help undoc' lists the undocumented commands.

grab [source]

grabs a single frame by unfreezing the video input for exactly two fields. This allows external compositing with the A60 output, normally when selecting video in this case the machine would just feedback.

source in this case is 1 or 2 for inputs 1 and 2 (defaults to one)

Installing an A60 on a Unix Network

These notes apply in particular to SunOS, there may be local variations.

Installation should be simply a matter of adding the Internet address chosen for the A60 to the file /etc/hosts and then entering this address on the A60 control panel.

For these changes to the host machine you will almost certainly require superuser privilege. This is probably the point to contact your local Unix Guru.

/etc/hosts

Edit the /etc/hosts file (note that root normally has a # as a prompt)
In the following notes "ourhost" is a fictitious example hostname.

ourhost# **vi /etc/hosts**

Entries in the hosts file have the following format : 192.9.200.5 a60

Everything to the right of a'#' is ignored as a comment. Lines consist of two or more fields separated by whitespace (tabs or spaces). The first field on a line is the internet address of the host. Following normal convention it is expressed as four decimal numbers separated by dots. The second field on a line is the hostname. Any subsequent names on the line are aliases for the same machine (normally local abbreviations or nicknames).

Setting the A60 Internet Address

The A60's Internet address is set via the Miscellaneous Menu on the control panel. To select the appropriate menu from the "Norm Play" home menu type **7 4 <Menu>**.
Then enter the address in two halves as pairs of bytes separated by a dot.
eg **192.009 <enter>**
200.005 <enter>
Note that leading zeroes have to be included.

Setting the A60 Hostname

The default A60 hostname is "a60" which is fine unless you have more than one. To make life easier it is possible to change the hostname on the A60 so that it is obvious which machine has been logged into. Simply log in to the a60 using Telnet or rlogin and issue the hostname command to set a new name.

```
ourhost% rlogin a60
Abekas A60 Remote Login
a60% hostname fred
fred% ^D
ourhost%
```

Special notes for Yellow Pages

Yellow Pages

If there is more than one Workstation connected to the network running NFS the chances are that the network service called the Yellow Pages will be running. The Yellow Pages allow all the machines on the network to share the same configuration tables, especially things like host names, password and account information. This simplifies the task of maintaining the system wide databases and means they only need to be updated in one place.

The existence of the Yellow Pages can be determined by typing the command *ypwhich* Unix should respond with the name of the yp server.

```
ourhost% ypwhich
ypwhich : ourhost is not running ypbind
```

Says that the Yellow Pages are not running. (ypbind is the name of the program that accesses the YP service)

```
ourhost% ypwhich
yphost
```

Says the Yellow Pages are originating from the host called 'yphost'

When the yellow pages are running, network nodes only consult their own tables at boot time, after this requests for system configuration such as hostnames are provided by the yellow pages service.

The new hostname need only be entered on the YP server machine. It then has to be "pushed' out to all the other clients.

First login on the YP server machine either over the net or by actually walking over to the YP server.

```
ourhost% rlogin yphost -l root
```

Update the /etc/hosts file as described above.

Now we have to update the dynamic version of the host table :

```
yphost* cd /etc/yp
yphost* make hosts
Updated hosts
Pushed hosts
```

/etc/ethers

There is a further refinement for certain networks that allows the A60 to ask the network what its' Internet address is by way of the Reverse Address Resolution Protocol (RARP). This requires an addition to the file /etc/ethers .

Note that this step will not usually be necessary since the A60 stores its Internet address in Non-Volatile RAM. It only uses RARP if the RAM contents are lost or the Internet Address is manually set from the control panel to 0.0.0.0 and the machine is restarted twice.

To find out if a host supports RARP try the following:

yphost* **ps -ax | grep rarp**

it should show you all the processes with anything to do with RARP apart from the line that says "grep rarp" (which is part of the command you just typed) there ought to be a mention of /usr/etc/rarpd if the RARP Daemon (the process that catches RARP requests) is running. If this is the case you can add the Ethernet address of the A60 to the /etc/ethers .

yphost* **vi /etc/ethers**

A typical entry in ethers is :

```
0:0:76:60:FF:FF    a60
```

Again if the Yellow Pages is running you have to force it to update its' copy of the 'ethers' file.

yphost* **cd /etc/yp**
yphost* **make ethers**

If the Internet address on the A60 is set to all zeros the A60 will attempt try to find its address from the network when it is rebooted twice. If you watch the Internet Address Menu it should show the correct address when it finds it.

If the Ethernet address of one of the Internet hosts changes you might have to flush the old entry from the ARP tables in the kernel. This can be achieved by using the arp command.

ourhost* **/etc/arp -d a60**

This deletes the table entry and forces the host to use ARP the next time a connection is required, therefore obtaining the latest Ethernet address.

Implementation Notes

The following notes list the specific aspects of the implementation of TCP/IP on the A60.

Ethernet Address

The unique Ethernet Address for each machine is set in Software PROM for the 68000 (top 6 bytes of the 27256 at location 7C on the computer card).

A60's have an officially allocated Ethernet address range 00:00:76:60:XX:XX the last two bytes being determined by the Computer Card Serial Number. The Ethernet address of a particular card should be engraved on the left hand side at the front of the card.

Note that if the Computer card is swapped the Ethernet Address is likely to be different.

The A60 is compatible with IEEE 802.3 or 10 Mbit/s Ethernet V2 - the only difference is the transceiver cable grounding arrangements.

The A60 doesn't care if the transceivers generate SQE - it's just ignored, however in practice it has been found that some transceiver/cable combinations have produced problems that went away when SQE was disabled.

Address Resolution

The A60 supports both the Address Resolution Protocol (ARP) and Reverse ARP.

The A60 will attempt to use RARP in the event that the machines' Internet address is set to 0.0.0.0 and a power-on reset occurs twice. This will either be the result of a really cold start - eg the Non Volatile RAM in the machine has been trashed or if the address is set to 0.0.0.0 from the control panel and the Computer Reset button is pressed twice.

RARP will only work if someone out there is serving up Ethernet/Internet address mappings (such as the rarpdaemon on a Sun which refers to the /etc/ethers file)

If there is no reply the Internet address will remain at zero.

IP

The A60 IP is not capable of re-assembling fragmented packets.

The A60 sends ICMP protocol unreachable for services it doesn't support such as UDP.

Incoming ICMP information is not recognized.

Ought to support GGP simple echo reply but it's never been tested.

Don't think we support 4.2 BSD/VAX Trailer Encapsulation whatever that is.

If the Least Significant Byte of the Internet address is configured as zero the ethernet drivers will use the broadcast address. I'm not sure why we do this any more.

TCP

The A60 TCP initial sequence number is always zero.

A60 TCP does not check security or precedence level of packets.

Since the A60 is not really a shareable resource it only supports one connection per socket. Any subsequent connection attempts will be ignored.

When a TCP packet is not acknowledged, the A60 will resend the packet up to twenty times with increasingly larger intervals before aborting the connection.

TCP Seg size option of 1460 is written out and the A60 responds appropriately by honoring incoming SEG SIZE option up to 1460. The default segment size is 512.

PSH flag is now set for each packet sent - Pyramid systems seem really sensitive to this.

Telnet

None of the Telnet options are supported. The A60 will just respond courteously with the appropriate negative response
eg WILL->DONT WONT->DONT DO->WONT etc

Ought to issue the Telnet Option "Suppress go ahead" especially since we don't issue Go Aheads !! but don't as yet.

A60 Telnet will accept CR, LF or CRLF as a line terminator.

The HP 9000/300 workstation has problems with A60 Telnet - <CR> doesn't locally echo as <CRLF> use <LF> instead.

Symbolics systems seem to think that remote echo is a default option for Telnet - it isn't in the book we have.

FTP

We attempt to minimize the password and accounting formalities by replying with a User OK after just a user name. Some applications at the other end still insist on providing a name and password however facilities such as the Unix '.netrc' file can smooth the FTP login sequence. The A60 isn't a real computer and there just wasn't space to keep track of names and access permissions.

Third party FTP transfers (controlled from somewhere different from the data connection) have never been tested.

When using FTP most applications require that TYPE IMAGE is specified in order to stop the local host expanding Carriage Returns into Carriage Return-Line Feed sequences. Although these aren't valid CCIR 601 video values (so they shouldn't appear in the file) it still takes the host time to filter the data.

Hangs up at the moment if the Remote client restarts and our end is still open. It's OK if we restart with the other end still up since the next packet sent from the other end will most likely be out of sequence and illicit a reset from our end.

rcp

A60 rcp can't cope with wildcard transfers eg the command

rcp *.rgb a60:

will return an error message

rlogin can be terminated by <control> D or <control> C In rlogin either ASCII back-space or delete have the same effect.

rcp actually uses the rsh port

General

Segments that have been set "record lockout" from the A60 control Panel appear to the file transfer utilities as write protected files.

Remote Control through the Login utilities is locked out while a file transfer is in progress.

When the A60 SCSI Port is in use the Ethernet services are not available

The A60 is still liable to hang up if the remote end of a connection goes away without any warning - it keeps the connection open and refuses any further attempts to connect to that port, even from the same host. At the moment the only way round this is to issue a reset command over rsh, rlogin or Telnet (assuming they are not all hung at once). The reset command will clear out all existing connections.

Filenames (or frame numbers) can have all sorts of directory junk on them - the parser works back from the far end of the name - eg optional extension, plus for second field, frame number, f for field.

Copyright Notice

Release Notes

1.05

fixes the 'slow mode' that occasionally occurred on busy Nets
provides extra debug for FIN states, RARP and transmit failures.

At the higher level File transfers now restore the original display mode
new commands: Disable, Enable, Protect, Unprotect and help.

1.06

fixes MIT PC problem Raw transfer ending mid-packet
Loop command with In -- Out is now trapped as illegal
Close Wait state actually waits - no longer sends RST
Doesn't send ACK unless some data arrives (fewer Resets)
Ether statistics eg lost packets available from rlogin
Max TCP Segment Size now offered as 1460 bytes (was 1024)
Transfer From A60 in RGB for 1460 byte packets fixed - upset rlogin
rcp to A60 non existent file handles error message correctly
rcp originating from A60 (rcp a60:0 host:file) - now trapped properly
Odd byte packet start addresses now OK
Tidied up Closing states FIN now retransmitted
handles time_wait state properly, waits to catch any ACKs
Locked out play commands during Transfer (NB keybd still active)
- No longer left in freeze after record command
Buffer wrap around problem - FTP MPUT (too many commas error !!)
also occasionally missed frame in the middle of long MPUT
Raw ends before all up to date esp from PC
Record command now shows next frame rather than In point
Test Patterns drawn a line higher in the store (1st line was black)
debug command with no param returns current setting

1.07

RGB conversion doesn't blow up with values less than 16
SYN's rejected on open connections
SYN's ignored during Time-Wait state
Window less than zero problem fixed
FTP waits longer for its' data connection
Chunks and reassembly cleared between connections

1.08

fixed Address error if RGB read allowed to have odd length
Retry times are now exponential eg wait twice as long each time
Window and retry time can now be tweaked from debug
TCP debug per tcb - includes # of times it gets ahead

1.09

Reset over rlogin clears framestore lock
NOOP FTP command fixed
TYPE L8 accepted in FTP

1.10

Help for undocumented commands
Help parameters can be shortened
Interpolator turned off if Bars are drawn

1.11

fixed bug in bug fix for non null sent by rcp
added PSH to all packets to keep Pyramid happy
New commands for paint systems, input, freeze and recrel

1.12

fixes crash when offered window is greater than 1460 - again!

1.13

fixes a couple of SCSI problems
extended commands didn't expect the correct number of command bytes
undid an old bodge to support Kennedy s/w 241-002 and greater

1.14

fixes another SCSI problems - hangups with a Sun 3/260
now have SCSI debug - select debug level 9
also fixes 'slow' mode ethernet - finally fixed those buffers and interrupts for good

Troubleshooting

Some of these things are probably covered elesewhere in the user manual but if like me you only turn to the manuals when something appears to be broken this might save you some reading.

Common Problems

Ethernet cable

Is the machine correctly connected to the ethernet cable ?

Assuming Ethernet activity can be generated from another host eg attempt to 'ping' the A60 repeatedly.

There is an LED on the computer card (second from left) that shows Ethernet activity Some Transceivers (eg Cabletron) also have transmit, receive and collision LEDs which are of great assistance at this stage.

Receive LEDs permanently 'on' on both the Transceiver and the Computer Card - Transceiver not connected to cable.

Collisions with every receive - cable incorrectly terminated.

Receive at transceiver but not at Computer - Problems with the Transceiver cable.

There have been some strange problems related to particular Tranceivers - if you are experiencing problems such a a lot of retrys and you have more than one type available try a different one . If possible use a transceiver with SQE turned off.

Ethernet Address

Has the Ethernet Address changed ?
In some cases where the software or the computer card is changed the Ethernet address might be different. Some systems retain hostname - ethernet address tables thatneed to be flushed in the unlikely event that the address changes. (see the last paragraph of the preceeding section on Intasllleing on a Unix network - /etc/ethers.

Internet Address

Is the Internet Address set up correctly ?

Assuming you have access to a 'ping' program and that the Internet address has been entered on the Control panel. Note that ping is sometimes hidden away in /etc or /usr/etc.

The Internet address should have the same network address as the rest of the hosts on the same local net. The network address is the top byte of the internet address if the top byte is less than 128, the top two bytes if the top byte is greater or equal to 128 and less than 192 or the top three bytes if the top byte is greater or equal to 192. If the Network part of the address is not the same the remote host will think the A60 is on a different network.

Most pings will have to start by using ARP to find the Ethernet Address of the A60.

There are two LEDs on the front of the computer card. **CSEN** shows ethernet carrier sense and should flicker with any network activity, **DMA** shows the Ethernet Controller is driving the 68000 bus indicating that packets are being transmitted or received by the A60.

If Ping fails more than about 1 percent of the time, unless the network is unusually heavily loaded this sounds like a noise or grounding problem. The A60 does suffer spurious collisions on transmit if the Transceiver cable is improperly shielded.

68000

Is the Motorola 68000 that controls the A60 Ethernet Hardware running ? If the 68000 has hung the A60 will be unable to write test patterns into the store but the disk transport functions may appear to work OK. - Press the Reset Switch on the Computer card at the right hand side near the front. - Check for bent pins on the 68000 PROMs 1C-7C

Disks

Are the disks up ? If communications appear to be OK but the A60 won't record or play, the disks might be spun down, there is a toggle switch on the front of the Computer card which should be to the right for normal operation.

Z80's

Has one of the Z80s crashed ? If the HLC or LLC goes down the A60 will not respond to commands from the control panel. The Status display may be showing an illegal timecode or may be corrupted. Operation of the Z80's is closely coupled to the incoming reference sync. If this is removed or replugged they may become confused. If the HLC is not running the whole machine hangs up and all the lights on the control panel stay on. If the Reset button fails to correct this condition try clearing the Non-Volatile RAM by changing the position of the first DIP switch at 18k on the computer card.

Reference Syncs

Is the external reference selected correctly ? If the disks are making strange noises trying to lock to a non-existant sync switch the Internal/External Sync select to Internal (the right).

RGB Transfers

If the data comes out stepped diagonally as a result of an FTP check that TYPE IMAGE was specified before the send (some FTP programs at the host end expand out CRLF sequences in the default ASCII mode !!)
If YUV data can be transferred OK but RGB comes out pink and green:
Is the file extension being specified correctly ?
Finally if all the data comes out green there may be problems with the MAC chip.

Colorized Video

If the computer card is changed and the existing material on the disk is suddenly weird colors you probably forgot to change the Flaw Map proms (19E and 23E) to the new computer card.

Field Service Call

In the event that you decide to call Abekas Field service for an Ethernet related problem it helps if you have the following information to hand.

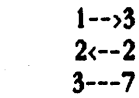
Software version numbers (see the diagnostic menu 4)
System configuration: Types of host computers/ Operating system
Network Configuration, how many hosts, is it busy?
Application used eg Rlogin, FTP, homegrown rcp
Which direction were you transferring data ?
Was it an RGB or YUV transfer
Did the connection hang up or the A60 crash (can you still draw color bars)

Doing your own Debugging

If you are really keen there is copious debug information generated by the A60 as packets arrive.

Debug Port

The A60 Computer has a RS232 serial port on the front of the Card for debug It runs at 9600 baud and only sends data it does not receive it. The connection is through the 5 pins on the right-hand end of the card The right most pin is pin 1 which is the transmit data from the A60 and the center pin, pin 3 is ground. Pin 2 would be receive if it did anything.



A60 Debug -- RS232 (DTE) 25 way DType

Debug Mask

The hex weightings of the bits in the debug mask are as follows:

2000	FIN debug - Turn on TCP debug as connection closes
1000	framestore addressing info
800	TCP window info (turns on TCP when packets are out of sequence)
400	Telnet
200	IP
100	TCP
80	Application eg FTP or rsh
40	Timeouts
20	Ethernet
10	Protocol Errors
8	Network Errors
4	Info messages
2	Dump packets (when IP debug is on as well)
1	Bughalt

Debug level can be set through any of the login connections by using the debug command with a hex argument.

Alternatively the debug level can be set from the control panel (menu 7 3) the equivalent debug setting in hex is given in parentheses.

0 Debug off	
1 Errors and TCP	(11C)
2 Errors TCP and Window	(91C)
3 Everything except Dump	(1FC)
4 Errors and Application	(09C)
5 Errors Telnet/Sockets and Application	(49C)
6 Errors Telnet/Sockets Application and TCP	(59C)
default Errors and TCP	(11C)

Note that with debug on the machine will run more slowly especially with the lower levels of debug where it has to print several lines per packet.
The A60 debug represents non-printing ASCII vales as a backslash followed by two hex digits.
The buffering for the debug port is fairly crude so if there are large amounts of debug information being transmitted the buffer can wrap round and information can become garbled.

High Level Debug

If the Telnet/Socket debug bit (400) is set the following messages will be generated

```
***** "\00"
***** "simon\00debug 80\0A"
```

Lines starting with five stars show what arrives on a packet by packet basis. The text is enclosed in double quotes and any non printing characters appear as backslash and two hex digits.

```
*** Open
*** RSH EXIT

TELNET recv opt: DO SUPPRESS-GO-AHEAD
```

If the Application debug bit (80) is set the following messages will be printed

Lines starting with three stars show significant events such as opening and closing connections.

```
RSH EXEC GOT FNAME
RSH EXEC play

TELNET EXEC goseg
```

TCP Debug

Port Numbers

20 FTP Data
21 FTP Command
23 Telnet
513 rlogin
514 rsh and rcp

These are the 'well known' port numbers used by the A60

TCP 1022 send [4] 102.9.200.3 1021 (tcb 4) Seq 33741057 Ack 1 Win 4096 SYN ACK

Intitially TCP debug describes each packet by the local port number and the foreign Internet Address and port. Once Established the TCP connection is only referenced by tcb (tcp control block) number.

TCP (4) send [0] Seq 1 Ack 33741058 Win 450 PSH ACK

IP Debug

ipdemux got pkt[120] prot 6 from 192.9.200.1

On the input side IP prints a message per packet with the length and the protocol number. Relevant protocol numbers are as follows:
TCP 6
ICMP 1
GGP 3

inwrite pkt[25] prot 6 to 192.9.200.1 route 192.9.200.1

Ethernet Debug

Ethernet level debug takes the following form:

ET_SEND: p[45] -> 192.9.200.1

ET_DEMUX: got pkt[60] buf(0) type IP

ET free buf(0)

Recognized packet types

IP 0800
ARP 0806
RARP 8035

A ping should look like this. First there is an ARP request for the A60's ethernet address, then an ICMP echo request.

```
ET_DEMUX: got pkt[60] type ARP
ET_SEND -> 08:00:20:01:FF:9C
et free buff(4)

ET DEMUX got pkt[98] buf(5) type IP
ipdemux: got pkt[64] prot 1 from 192.9.200.1
ICMP: p[64] from 192.2.200.1
ICMP: echo reply to 192.2.200.1
inwrite pkt[64] prot 1 to 192.9.200.1 route 192.9.200.1
ET_SEND: p[64] -> 192.9.200.1
```

Some packets will be discarded - ARP requests that are not intended for this machine and broadcast packets from unsupported protocols such as XNS.

Glossary

ARP
Address Resolution Protocol - used to obtain Internet to Ethernet address mappings when they are needed rather than maintain a static list on each host.

ARPA
Advance Research Projects Agency, US Government agency responsible for developing TCP/IP family of protocols.

AUI
Attachment Unit Interface - the long way of saying transceiver cable. Normally limited to 50 meters and carries twisted pair differential signals for transmit, receive and collision detection. The AUI cable also carries 12v power for the Transceiver.

Bridge
Bridges are generally connections between Networks of the same type at the Data Link Layer.

Broadcast Packet
an Ethernet packet carrying the address FF:FF:FF:FF:FF:FF which will be received by all the hosts on the network.

CCIR 601
This is the Standard for digital video drawn up by the international committee it is also referred to as SMPTE RP125 and EBU 3246-E.

Cheapernet
an alternative form of ethernet that uses thin RG58 50-Ohm cable with BNC connectors and 'T' pieces at the transceivers rather than that normally used for Ethernet. This cable suffers from greater loss and the the cable run is typically limited to 185 m. In all other respects it is electrically compatible with 'Normal' Ethernet.

Client
The consuming (user) end of a client-server relationship.

Connection
the link between two specific ports on two Internet Hosts analogous to a telephone call being set up between two phones on a network.

CRC
Cyclic Redundancy Check - a sort of serial check sum that assures the integrity of a serial data stream by using some sort of polynomial feedback - basically a shift register and a few exclusive-or gates.

CSMA-CD
Collision Sense Multiple Access with Collision Detection - Describes the mechanism that allows several devices to share the single Ethernet cable.

Domain Names
A name addressing scheme that uses a hierarchy of domain names to describe the address of a remote computer in a similar way to a mailing address eg simon@master.abekas.COM

Ethernet V1 and V2
The original Ethernet standard was developed by Xerox and published jointly with DEC and Intel.

Ethernet Address
A 48 bit address conventionally written as 6 hex bytes separated by colons The IEEE hopes that there are enough addresses for every piece of ethernet equipment in the universe to have its own unique address. Typically vendors ship equipment with the Ethernet address contained in a small bipolar ROM.

Fragments
Sub divisions of Internet packets - sometimes necessary if different networks have different maximum packet lengths.

FTP
File Transfer Protocol

4:2:2
The Ratio of the sampling frequencies for components in Digital Video. Four Luminance samples for each two of the color difference signals.

Gateway
Gateways allow interconnection of different Networks at the Network layer.

IEEE 802.3
The IEEE standard for CSMA/CD Networks, forms part of the 802.x family of standards for Local Area Network Interfaces and Protocols.

Internet Address
A Four byte number representing the address of a host on the "Internet".

Jabber
A condition detected by a Transceiver to prevent locking up the network - If the output is active for more than 1/10 second the Transceiver should latch up and prevent further transmissions until the Transmit signal is inactive for at least 1/4 second.

Jam
A collision signal generated by a repeater so that the cable segments on both sides of it are aware that a collision has occurred.

Heartbeat
See SQE

Host
A computer that is a node on a network.

Hostname
The name by which a particular machine is known at the user interface level - normally associated with a Network address.

MAU
Medium Attachment Unit The official IEEE name for what mortals refer to as a Transceiver.

NFS
Network File System (Developed by Sun Microsystems)

NIC
Network Information Center - The central repository for all information regarding the development of the ARPAnet

OSI Layered Model
The much vaunted abstract model for the seven layer hierarchy of network protocols issued by ISO the International standards body. Still under development at the higher layers.

Ping
Program that uses the ICMP Echo Request facility to verify the connections between two machines. Note that this only verifies correct operation up to and including the Network layer or Internet Protocol.

Port Number
An addressing scheme within a host computer that allows more than one simultaneous connection to that computer.

Repeaters
Repeaters interconnect or buffer two sections of the same Network with very little intelligent signal processing.

RFC
Request For Comment - the main instrument of the ARPA Community a sort of Network Memo. Often quoted in references e.g. RFC 793 is the latest description of TCP. They are available from the Network Information Center (see NIC above).

Server
A host or node on the network that provides a service, eg a File Server provides a File system for Diskless nodes.

SQE
Signal Quality Error - some Ethernet Transceivers generate a collision signal immediately after each transmission. This permits the Ethernet Interface to verify that its collision detection circuitry is functioning correctly.

Socket
A socket is an abstraction in BSD Unix for the interprocess communications primitive the known as the pipe. A socket can be opened in much the same way as a file would be opened.

Telnet
ARPA remote terminal/login program.

TFTP
Trivial File Transfer Protocol - built on UDP - simple to implement.

10BASE5 and 10BASE2
IEEE names for Ethernet and Cheapernet respectively, abbreviated from **10**MHz **Base**band **5**00m and **2**00m maximum length. (watch for a twisted pair spec)

Thin-LAN or Thin-net
see Cheapernet

Transceiver
The box which attaches the Host computers' Ethernet Interface to the cable. The best ones use a spike arrangement to non-intrusively attach to the cable (eg without cutting it). The transmit and receive signals are coupled onto the cable through isolating transformers. The transceiver is also responsible for detecting collisions.

UNIX™
is a Trademark of AT&T Bell Laboratories! Available in two main flavors (at the time of writing) BSD 4.2 and ATT system V, others based on the PC platform include SCO, Microport and Xenix. The Berkeley Software Distribution (BSD) version is the one that first supported the ARPA protocols, most popular Unix Systems eg SunOS, HP-UX and DEC Ultrix are a mixture of the Berkeley and AT&T.

UDP
stands for the User Datagram Protocol, a datagram is a stand alone packet with no can be transmitted over the network with no implied sequence or guarantee of delivery. UDP is an alternative to TCP at the Transport Level NFS and Yellow Pages are built on top of it.

X.25
Public Wide Area Packet Switched Network Standard.

Yellow Pages
The Yellow Pages is a network service that provides network wide Password and system information from a single database. Written by Sun Microsystems but available on many systems.

YUV
the Native data format of the A60, an alternative encoding to RGB for component video and one that is more suited to the limited bandwidth of Broadcast TV. The Y channel is luminance and the U and V are color difference components B-Y and R-Y respectively. The color difference signals are normally stored at half the sampling frequency of the Luminance.

Example Programs

The following programs are examples of remote control and file transfer from within a program on a Unix system using the *socket* interface.

The first is a simple SunView application intended to demonstrate use of the Berkeley Socket interface and remote control of the A60 over Ethernet.

It consists of three files, demo.c the Ethernet interface to the A60, panel.c a simple Suntools panel, and a60.icon an icon used by the program. To compile it you have issue the following command assuming you have access to the appropriate libraries.

% cc demo.c panel.c -lsuntool -lsunwindow -lpixrect -o demo

There can only be one connection (to A60 rlogin) open at a time so if the window does not appear the A60 rlogin connection must already be open.

The program makes no attempt to interpret or dispose of any output from the A60 this is just being buffered up. This should not prove to be a problem since there is no echo unless the rlogin was opened up with the string "\0simon\0simon\0sun/9600\0". Rlogin turns off echo if the terminal speed parameter is omitted.

The second program demonstrates transfer of an RGB image directly into the A60 from a program, simulating the operation of the **rcp** command. It does not use the SunView interface. In the form presented it has very limited use since it always transfers the image to a hard-coded frame number. If the image is to be in YUV format the file extension specified in the command string and the transfer length will be different.

Note that this program will not copy to a normal Unix host because it bypasses the normal access permissions required by Unix safe the knowledge that the A60 does not require a reserved originating socket number below 1024. The program normally has to be running as root to bind to a privileged socket.

demo.c

```
/* demo.c - quick lash up to show remote control of A60
 * - from within a Suntools app - using TCP/IP
 * Copyright ©1987 Abekas Video Systems Inc.
 *
 * Based on an example proggys
 * in the IPC Primer in "Networking on the Sun Workstation"
 */

#include <stdio.h>
#include <netdb.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

char login_str[] = "\0simon\0simon\0sun\0";
char play_fwd_str[] = "play\n";
char play_back_str[] = "play -1\n";
char stop_str[] = "stop\n";

int sd;                 /* global socket descriptor */

open_connection()
{
struct sockaddr_in skt;
struct servent *rlogin_service;
struct hostent *a60;
int lport;
char c, tmp_str[80], str[80];

if((rlogin_service = getservbyname("login", "tcp")) == NULL)
   {
    fprintf(stderr, "demo : tcp: unknown service\n");
    exit(1);
   }
if((a60 = gethostbyname("a60")) == NULL)
   {
    fprintf(stderr, "demo : a60 : unknown host\n");
    exit(1);
   }
```

```
bzero((char *)&skt, sizeof(skt));
bcopy(a60->h_addr, (char *)&skt.sin_addr, a60->h_length);
skt.sin_addr.s_addr = INADDR_ANY;

if((sd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
   {
    perror("demo : socket");
    exit(3);
   }

bzero((char *)&skt, sizeof(skt));
bcopy(a60->h_addr, (char *)&skt.sin_addr, a60->h_length);
skt.sin_family = a60->h_addrtype;
skt.sin_port = rlogin_service->s_port;
if((connect(sd, (char *)&skt, sizeof(skt))) < 0)
   {
    perror("demo : connect");
    exit(3);
   }
printf("OK so far\n");

write(sd, login_str, sizeof(login_str));
}

close_connection()
{
close(sd);
}

play_forwards()
{
write(sd, play_fwd_str, strlen(play_fwd_str));
}

play_backwards()
{
write(sd, play_back_str, strlen(play_back_str));
}

stop()
{
write(sd, stop_str, strlen(stop_str));
}
```

panel.c

```
/* panel.c -  quick lash up to show remote control of A60
 * - from within a Suntools app - using TCP/IP
 * see Sunview programmers guide for examples
 * Copyright (c) 1987 Abekas Video Systems Inc.
 */
#include <suntool/sunview.h>
#include <suntool/canvas.h>
#include <suntool/panel.h>
#include <stdio.h>
#include <math.h>

static short icon_image[] = {
#include "a60.icon"
};

DEFINE_ICON_FROM_IMAGE(a60_icon,  icon_image);

Frame frame;
Canvas canvas;
Panel a60_panel;
Panel_item play_fwd_button, stop_button, play_back_button;

Pixwin *pw;
Pixfont *font, *bold;

static Notify_value catch_closes();

/* calls five routines from demo.c
 * these first three are called when the appropriate button is pressed */

extern void play_backwards();
extern void stop();
extern void play_forwards();

/* also calls open_connection and close_connection as the window
 * is opened and closed */
```

```
main()
{
/* open fonts */
font = pf_open ("/usr/lib/fonts/fixedwidthfonts/screen.r.12");
bold = pf_open ("/usr/lib/fonts/fixedwidthfonts/screen.b.12");

/* create frame */
frame = window_create(NULL, FRAME,
                    FRAME_LABEL, "demo - A60 panel",
                    FRAME_ICON, &a60_icon,
                    0);

a60_panel = window_create(frame, PANEL,
                    0);

play_back_button = panel_create_item(a60_panel, PANEL_BUTTON,
                    PANEL_NOTIFY_PROC, play_backwards,
                    PANEL_LABEL_IMAGE,
                        panel_button_image(a60_panel, "<<", 4, bold),
                    0);

stop_button = panel_create_item(a60_panel, PANEL_BUTTON,
                    PANEL_NOTIFY_PROC, stop,
                    PANEL_LABEL_IMAGE,
                        panel_button_image(a60_panel, "STOP", 6, bold),
                    0);

play_fwd_button = panel_create_item(a60_panel, PANEL_BUTTON,
                    PANEL_NOTIFY_PROC, play_forwards,
                    PANEL_LABEL_IMAGE,
                        panel_button_image(a60_panel, ">>", 4, bold),
                    0);

window_fit(a60_panel);
window_fit(frame);

notify_interpose_event_func(frame, catch_closes, NOTIFY_SAFE);

open_connection();
window_main_loop(frame);
}
```

```
/* this routine intercepts events and catches open and closes */

static Notify_value catch_closes(frame, event, arg, type)
Frame frame;
Event *event;
Notify_arg arg;
Notify_event_type type;

{
int was_closed, now_closed;
Notify_value value;

was_closed = (int) window_get(frame, FRAME_CLOSED);

value = notify_next_event_func(frame, event, arg, type);

now_closed = (int) window_get(frame, FRAME_CLOSED);

if(was_closed != now_closed)
   if(now_closed) close_connection();
   else open_connection();

return(value);
}
```

a60.icon

```
/* Format_version=1, Width=64, Height=64, Depth=1, Valid_bits_per_item=16
 */
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x07FF,0xFFFF,0xFFFF,0xFFC0,0x0622,0x2222,0x2222,0x2240,
  0x0444,0x4444,0x4444,0x4440,0x05FF,0xF911,0x1115,0x3140,
  0x04FF,0xF888,0x888A,0x88C0,0x06FF,0xFA22,0x223E,0xB240,
  0x0444,0x445F,0xFFC5,0x7440,0x0511,0x111F,0xFF90,0x1140,
  0x0488,0x889F,0xFF88,0x88C0,0x06E3,0xBF3F,0xFFA2,0x2240,
  0x04E7,0xDF5F,0xFFC7,0xE440,0x05F3,0x9F1F,0xFF97,0xF140,
  0x04EB,0x9F9F,0xFF8F,0xF8C0,0x06E3,0xBF3F,0xFFBF,0xFE40,
  0x04E7,0xDF5F,0xFFDF,0xFC40,0x05F3,0x9F1F,0xFF9F,0xFD40,
  0x04EB,0x9F9F,0xFF9F,0xFCC0,0x06E3,0xBF3F,0xFFBF,0xFE40,
  0x04FF,0xDF5F,0xFFCF,0xFC40,0x05FF,0x9F1F,0xFF97,0xF140,
  0x04FF,0x9F9F,0xFF8B,0xE8C0,0x0622,0x2222,0x2222,0x2240,
  0x0444,0x4444,0x4444,0x4440,0x0511,0x1111,0x1111,0x1140,
  0x07FF,0xFFFF,0xFFFF,0xFFC0,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0083,0x8700,0x0000,0x0000,0x0084,0x4880,0x0000,
  0x0000,0x0144,0x0880,0x0000,0x0000,0x0144,0x0980,0x0000,
  0x0000,0x0147,0x8A80,0x0000,0x0000,0x0224,0x4C80,0x0000,
  0x0000,0x03E4,0x4880,0x0000,0x0000,0x0224,0x4880,0x0000,
  0x0000,0x0223,0x8700,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,
  0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000,0x0000
```

mycp.c

```
/* mycp.c example of a homebrew rcp
 * this version works for rgb and always writes to the same frame !
 * © 1988 Abekas Video Systems Inc.
*/

#include <stdio.h>
#include <netdb.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

char login_str[] = "\0simon\0simon";
char command_str[] = "rcp -t 0.rgb";                    /* this ought to be variable */
char file_str[] = "C0666 1049760 mypiccy\n";            /* different length for yuv */
char close_str[] = "\0";

int sd;                 /* global socket descriptor */

main(argc, argv)
int argc;
char **argv;
{
int i, j;
FILE *piccy;
char line[720*3];
open_connection();
piccy = fopen(argv[1], "r");
for(i=0; i<486; i++)
   {
    fread(line, 720*3, 1, piccy);
    write(sd, line, 720*3);
   }
close_connection();
}

close_connection()
{
char tmp_str[20];
int n;

write(sd, close_str, sizeof(close_str)-1);
n = read(sd, tmp_str, 20);
close(sd);
}
```

```
open_connection()
{
struct sockaddr_in skt;
struct servent *rlogin_service;
struct hostent *a60;
int lport;
char c, tmp_str[80], str[80];
int n;

if((rlogin_service = getservbyname("shell", "tcp")) == NULL)
   {
    fprintf(stderr, "demo : tcp: unknown service\n");
    exit(1);
   }
if((a60 = gethostbyname("a60")) == NULL)
   {
    fprintf(stderr, "demo : a60 : unknown host\n");
    exit(1);
   }
bzero((char *)&skt, sizeof(skt));
bcopy(a60->h_addr, (char *)&skt.sin_addr, a60->h_length);
skt.sin_addr.s_addr = INADDR_ANY;

if((sd = socket(AF_INET, SOCK_STREAM, 0)) < 0)
   {
    perror("demo : socket");
    exit(3);
   }
bzero((char *)&skt, sizeof(skt));
bcopy(a60->h_addr, (char *)&skt.sin_addr, a60->h_length);
skt.sin_family = a60->h_addrtype;
skt.sin_port = rlogin_service->s_port;
if((connect(sd, (char *)&skt, sizeof(skt))) < 0)
   {
    perror("demo : connect");
    exit(3);
   }
write(sd, login_str, sizeof(login_str));
n = read(sd, tmp_str, 20);
printf("read %d\n", n);
write(sd, command_str, sizeof(command_str));
n = read(sd, tmp_str, 20);
printf("read %d\n", n);
write(sd, file_str, sizeof(file_str)-1);
n = read(sd, tmp_str, 20);
printf("read %d\n", n);
}
```

Appendix

This information is provided for those who intend to write their own applications to 'manually' access the file transfer and remote control services on the A60.

TCP/IP Packet Dump

The following is a complete dump of a packet in case anyone out there is interested.

Physical
| Link
| | Network
| | | Transport
| | |

AA AA AA AA AA AA AA AB **Ethernet Preamble and sync**

00 00 76 60 FF FF **Destination Addr**
08 00 20 01 FF 9D **Source Addr**
08 00 **Type**

45 00 **Version, Header len, Type of Serv**
00 35 **Length**
FE 7D 00 00 **ID, Fragment Stuff**
0F 06 **Time to live, Protocol**
9D 2B **Header Checksum**
C0 09 C8 01 **Source Address**
C0 09 C8 05 **Destination Address**

21 83 **Source Port**
00 17 **Destination Port**
18 4A 0E DB **Sequence**
00 00 00 3B **Acknowledge**
50 18 **Flags**
10 00 **Window**
AA CF **Checksum**
00 00 **Urgent Pointer**

68 65 6C 6C 6F 20 77 6F 72 6C 64 0D 0A **Data**

2A 2B 49 8D **Ethernet CRC**

FTP Implementation

The following notes are intended to show the various states and messages of the Abekas FTP server.

Defaults

As suggested in the ARPA FTP Paper these are the defaults assumed by A60 FTP.

```
ftp->fport =      20;
ftp->lport =      20;
ftp->passive =    FALSE;
ftp->type =       ASCII;
ftp->format =     NON_PRINT;
ftp->byte_size =  8;
ftp->mode =       STREAM;
ftp->structure =  FILE;
```

Opening Message

```
220 Abekas A60 FTP (Hostname)
220 Abekas A60 FTP
```

Commands and Responses

The following are a list of all the commands supported and the possible responses. Some of the responses are as the result of internal events such as ABORT where the data connection was closed mid-transfer.

```
USER
  name > 20
  500 Username too long
  230 User OK
QUIT
  221 Closing control connection
PASV
  227 Entering Passive Mode. iii,iii,iii,iii,ppp,ppp
PORT
  error
  501 Host/Port spec not enough commas

  200 PORT spec accepted : host iii.iii.iii.iii port nnn
```

```
TYPE
   error
   501 TYPE : bad args
    ASCII
   NON_PRINT
      200 TYPE OK

   NON_PRINT
      TELNET
      CC
            504 can't cope with that TYPE

    IMAGE
   200 TYPE OK

    EBCDIC
   504 can't cope with that TYPE
   LOCAL
       8
            200 TYPE OK
       else
            504 can't cope with that TYPE

MODE
   error
   501 MODE : bad arg
    STREAM
   200 MODE OK
    BLOCK
    COMPRESSED
   504 can't cope with that MODE
STRU
   error
   501 STRU : bad arg
    FILE
   200 STRU OK
    RECORD
    PAGE
   504 can't cope with that STRU
```

```
RETR
   argc != 2
   501 arg count
   name parse
   550 Bad File name
   data conn
   125 Data Connection already open
   150 OK here goes

   UNOPEN
   425 Can't open Data Connection
   ABORT
   426 Connection Closed Transfer Aborted
   LRESET
   426 Connection Closed Local Reset
   FRESET
   426 Connection Closed Foreign Reset
   NORMAL
   226 File Transfer OK

STOR
   name parse
   550 Bad File name

   data_conn
   125 Data Connection already open
   150 OK here goes

   UNOPEN
   425 Can't open Data Connection
   ABORT
   426 Connection Closed Transfer Aborted
   LRESET
   426 Connection Closed Local Reset
   FRESET
   426 Connection Closed Foreign Reset
   NORMAL
   226 File Transfer OK

NOOP
   200 NOOP OK
else
   202 Command not implemented
```

File Names

The A60 is not case sensitive
The correct syntax for a 'filename' on the A60 is as follows:
optional leading slash or directory path (ignored)
'F' to indicate a field transfer (optional)
timecode 19.20 frame at 19 seconds 20 frames
or 19:20 second field
or 590 frame 590
or 590+ frame 590 second field
extension .RGB or .YUV (optional - defaults to YUV)

rlogin Implementation

In the following section "\0" indicates a null byte
"\001" indicates a byte of value 1
"\n" indicates ASCII Line Feed
"\r" indicates ASCII Carriage Return
"\b" indicates ASCII Back Space

(open connection to port 513)
--> \0 **(no standard error connection)**
--> simon\0simon\0 **(local and foreign username)**
--> vt100/9600\0 **(term/speed)**
<-- \0 **(user info validated)**
<-- Abekas A60 Remote Login\n
<-- a60%

User input is echoed character for character
Both backspace and delete characters do the same thing eg
--> \b **(backspace)**
<-- \b \b **(backspace space backspace)**

The Sun appears to leave out the /9600 bit when the connection doesn't expect echo eg when invoked by "rlogin host < script"

Venix running on a PC, sent \n\n instead of vt100/9600

Bibliography

Related Documents from Abekas

Abekas A60 External Control Protocol Manual
Abekas A60/A64 Digital Video Manual
Abekas A60/A64 SCSI Manual

ARPA Publications

DDN Network Information Center
SRI International, Room EJ291
333 Ravenswood Avenue
Menlo Park, CA 94025

DDN Protocol Implementations and Vendors Guide
A useful compendium of different TCP/IP applications
revised bi-annually

DDN Protocol Handbook (Three Volumes)

Particular RFC's of interest :

RFC 826 Address Resolution Protocol Nov 82 David Plummer Symbolics

RFC 903 Reverse Address Resolution Protocol June 84 Finlayson, Mann, Mogul, Theimer Stanford University

RFC 791 DARPA Internet Protocol Sept 81 J Postel (Editor)

RFC 792 DARPA Internet Control Message Protocol Sept 81 J Postel (Editor)

RFC 793 DARPA Transmission Control Protocol J Postel (Editor)

RFC 854 TELNET Protocol Specification May 83 Postel and Reynolds

RFC 959 File Transfer Protocol Oct 85 Postel and Reynolds

IEEE 802.3 LAN Standards
CSMA/CD Access Method and Physical Specifications
Oct 84
IEEE

Ethernet specs :

A LAN Data Link Layer and Physical Layer Specification
Version 2 Nov 82
DEC, Intel, Xerox

Other Background Reading

Why is it that none of these texts were around when I started on this project !

Byte Magazine July 87

Scientific American October 87 P 136
Networks for Advanced Computing
Robert E Kahn

Unix Papers for Unix Developers and Power Users p307
Ethernet: A Unix LAN
Charles Spurgeon
The Waite Group/ Howard Sams & Co 87

Handbook of Computer Communication Standards Vol 3
Department of Defense (DOD) Protocol Standards
William Stallings, Macmillan 88

Internetworking with TCP/IP
Principles Protocols Architectures
Douglas Comer, Prentice Hall 88

TCP/IP Implementations

The following machines are known to support TCP/IP, in some cases it will depend on the operating system.

Digital Equipment Corp. VAX and microVAX running ULTRIX or BSD Unix

Hewlett Packard HP 9000 300 and 800 running HP-UX

Sun Microsystems

Apollo Computer

Pyramid

Cray

Symbolics

Silicon Graphics

Apple Computer Macintosh II with Ethertalk

For the remainder there are add ons available from the following vendors (Information based on entries in the DDN Protocol Implementations and Vendors Guide)

Excelan
Intelligent Ethernet Controllers for VAX, MicroVAX and IBM-PC. TCP/IP software support for RSX-11, VMS, Unix System V and MS-DOS
2180 Fortune Drive
San Jose, CA 95131
(408) 434-2300

Wollongong
TCP/IP software for HP 9000 series 500, VAX, IBM-PC
1129 San Antonio Road
Palo Alto, CA 94303
(415) 962-7200

3com Corporation
Ethernet Interfaces for the IBM-PC
1365 Shorebird Way
PO Box 7390
Mountain View, CA 94039
(415) 961-9602

FTP Software
TCP/IP software for the IBM-PC
PO Box 150
Kendall Square Branch
Boston, MA 02142
(617) 864-1711

Abekas Video Systems, Inc.

A Carlton Company

Abekas Video Systems, Inc.
101 Galveston Drive
Redwood City,
CA 94063

Tel. 415·369·5111
Telex. 592712
Fax. 415·369·4777