AppleTalk Remote Access: Sample V.32/Slower Script

Article Created: 15 June 1992
Article Last Reviewed:
Article Last Updated:

TOPIC -----

I have a new V.32 modem that I want to use with AppleTalk Remote Access (ARA), but there's no script on the application disk that I can use with it.

DISCUSSION -----

The following AppleTalk Remote Access script for the Telebit QBlazer modem is a good example of how to write a script for any given V.32 (or slower) modem.

Notes

- This discussion assumes that you have access to the documentation for AppleTalk Remote Access and that you are familiar with the command syntax and scripting basics.

- The command syntax is in the AppleTalk Remote Access Modem Scripting Language Guide. You can order the guide from APDA at 800-282-2732 (USA), 800-637-0029 (Canada), or 408-562-3910 (International).

- If you have access to AppleLink, look for related AppleTalk Remote Access script files in the Software Sampler folder.

- For more information, use "AppleTalk Remote Access and V.32bis" as a search string.

```
! "Telebit QBlazer Modem - 12/10/91" JFK
! Adapted from the 7/29/91 T1600 script that ships with ARA 1.0
! 12/10/91 JFK - The QBlazer script is only one line different from the T1600,
!                but in my grand style of "Give them so much more they choke,
!                and then you can run around them..." I added the
!                "Ton'o'comments".
!                And fixed the one line, of course...
```

```
! 12/11/91 JFK - Figured out that S51 needs to be set to 252 on the QBlazer
!                instead of 255 like on the T1600 and T3000.
! 3/30/92  JFK - Added comments about the hang up sequence.
! 4/3/92   JFK - Added sbreak in hangup label, added S38 for ATH behavior,
!                moved some commands around for consistency.
!
@ORIGINATE
@ANSWER
!
! Talk to the modem at 9600 bps.  The QBlazer should auto-baud this
! unless the user has locked the port to a particular speed.  If it
! is locked to a different speed, the user will need to change that.
!
serreset 9600, 0, 8, 1
!
!  The idea here is to get the modem into a known state, and then change only
!  the registers that are necessary to support the connection.  Most of the
!  time AT&F will be sufficient, but some modems allow the user to change the
!  F0 parameters.  There isn't much that can be done to prevent this, but if
!  the modem has any pre-configured configurations that will set most of the
!  required parameters, use it.
!
! Recall the factory configuration
!
! AT&F0 set:
! S61=1 - Go into command mode when receiving break from DTE (see
!         @HANGUP for why the script cares about this).
!
!  Every time the script needs to send commands to the modem, the strategy is:
!  Clear all matchstrings, look for specific responses, and loop around a
!  couple of times.  Later in the script, certain loops pause 50-70 seconds,
!  such as when the script dials a number and is waiting for a connection.
!  Other times, the script pauses 3-5 seconds and loops around. When the script
!  is sending commands to the modem, it should expect to see a response within
!  a couple of seconds, so it's best to look quickly and exit with an error in
!  a reasonable amount of time so the users do not wait a for a long time
!  before they are notified that they may need to power-cycle/reset the modem.
!  When the script is dialing out over a telephone system or PBX, it needs
!  enough time to make a connection.  In short, if it's communicating to a
!  modem, loop in 3-7 second increments.  If the script is waiting for
!  something other than a modem response (like a completed connection or
!  terminal server) it may need 60-70 seconds.
!
!  If the defaults cannot be set, jump down to label 59, which exits and asks
!  the user to check out the modem.  If an  AT&F command will not be accepted,
!  the modem may be hung and needs to be manually reset.
!
settries 0
matchclr
@LABEL 1
matchstr 1 3 "OK\13\10"
write "AT&F0\13"
matchread 20
```

```
inctries
iftries 2 59
! Modem is not responding, reset and send a break
DTRClear
pause 5
DTRSet
SBreak
jump 1
!
!  The script was able to get the modem into a default factory state.  Now
!  set the basic hardware type configuration such as command echo, hardware
!  handshaking, and DTR control.  If the &F9 command had not set up handshaking
!  this is where it would be done.  It's not desirable to create one long
!  command string with everything on it because some modems cannot handle a
!  long command string, and long strings are harder to debug.  It's easy
!  to enter an incorrect S-register value.  For the most part, the following
!  commands are probably common across a lot of modems, but always look up the
!  commands in the modem manual.  For V.32 or slower modems, there should not
!  be any kind of flow control between the modem and the Macintosh.  This is a
!  different situation from V.32bis and faster modems (which require hardware
!  handshaking).
!
! Next, Set up the configuration: drop connection after losing DTR
! Turn off auto answer, command echo, and no DTE flow control.
!
! &D3   - DTR on/off resets modem
! S0=0  - Don't answer calls
! E0    - Turn command echo off
! S58=0 - No DTE flow control
!
@LABEL 3
matchclr
matchstr 1 4 "OK\13\10"
write "AT&D3S0=0E0S58=0\13"
matchread 30
jump 59
!
!  Now that the modem hardware & flow control parameters are set, make sure any
!  protocol negotiation is disabled, and issue any modem specific features
!  here.  Make sure that MNP4/V.42, and MNP5-10/V.42bis negotiations are
!  disabled.  By the way, some V.32/V.32bis modems have an option to disable
!  Trellis error control, which is part of the physical layer modulation.
!  This is not the same as MNP/V.42, and you do not want to disable it!
!
!  Make sure that the modem is configured so it does NOT require error control
!  to complete a link.  ARA 1.0 does all error correction/data compression in
!  software.  All ARA wants is the fastest raw data pipe it can get.  If the
!  script spends time trying to negotiate some error control, the modems and/or
!  Remote Access may time out.
!
!  Also note the S38 configuration.  It is noted later in the script that it is
!  desirable to ensure that the modem's buffer has transmitted all of its data
!  before it actually hangs the modem up.  This ability appears to be
```

```
!  implemented on a lot of modems.
!
!  This set of commands is going to be implemented differently on different
!  vendors modems.  In this example, Telebit uses S registers. Other modems may
!  use S registers (but different registers), or \ commands, or % commands; you
!  get the idea.  (Did I mention that you really, really want to have your
!  modem manual handy?)
!
!  It is important that the modem is configured so that it returns
!  the connected speed, NOT the DTE speed.  The script needs to know what the
!  real line speed is in order to set ARA's internal timers.  Some
!  modems don't have the option to display the line speed.  In that case the
!  performance of the connection may not be optimal.
!
! Next, disable MNP/error control, internal buffering, delay before
! disconnect, and issue extended result codes
!
! S180=0 - Turn off all error detection/correction (ARA does MNP and
!          compression itself.  It needs these turned off in the modem).
! S181=0 - Turn off DTE <-> line buffering if there is no error control.
!                           The idea is to have the Macintosh communicate
!                           with the modem at the line speed of the modem.
! S38=255  - Wait until the modem's buffer is clear OR the other modem
!            disconnects after an ATH is issued before dropping the line.
!            This is done to ensure that all/any data in the modem's buffer
!            has been transmitted to the remote modem before it disconnects.
!            If the remote connection does not receive the
!            disconnect packet (usually the last one sent) it could take
!            up to 45 seconds for the remote connection to timeout and
!            disconnect.
! X2    - Issue extended result codes.  This will display busy, connect XXX,
!         etc.  X2 will say "CONNECT XXX"  Where XXX is the line speed.  This
!         is so ARA can determine what speed the modems are communicating at
!         to set the serial port speed.
!
@LABEL 4
matchclr
matchstr 1 5 "OK\13\10"
write "ATS180=0S181=0S38=255X2\13"
matchread 30
jump 59
!
!  The modem should now be properly configured.  Now check to see if the user
!  has turned off the modem speaker.  If they have, send an additional command
!  to turn it off.
!
! If speaker on flag is true, jump to label 8.  Otherwise turn off the speaker.
!
@LABEL 5
ifstr 2 8 "1"
matchstr 1 8 "OK\13\10"
write "ATM0\13"
matchread 30
```

```
jump 59
!
! The modem is ready so enable answering, or originate a call.
!
@LABEL 8
ifANSWER 30
note "Dialing ^1" 3
write "ATS0=0DT^1\13"
!
!  Be aware that different modems will have different format strings
!  to return connection results.  You need to understand the different possible
!  strings and set this area (and then answer area at label 31) to the
!  appropriate value.  Also, remember that the modem was configured to return
!  the connect speed if possible (The X2 command up at label 5).  It's also
!  useful if the modem can return busy, no dialtone, etc. since the script will
!  be able to exit quicker and let the user know what is going on.
!
!  Also note that the script waits at the bottom of label 9 for 70 seconds,
!  rather than looping around.  Why?  Well, if the script re-issues the dial
!  command too soon, that would cause the modem to hang up.  At this point the
!  script should wait a reasonable amount of time for one of these strings to
!  return from the modem and take the appropriate action.
!
@LABEL 9
matchstr  1 11 "CONNECT 1200\13\10"
matchstr  2 12 "CONNECT 2400\13\10"
matchstr  3 13 "CONNECT 4800\13\10"
matchstr  4 14 "CONNECT 9600\13\10"
matchstr  5 15 "CONNECT FAST\13\10"
matchstr  6 50 "NO CARRIER\13\10"
matchstr  7 50 "ERROR\13\10"
matchstr  8 52 "NO DIALTONE\13\10"
matchstr  9 53 "BUSY\13\10"
matchstr 10 54 "NO ANSWER\13\10"
matchread 700
jump 59
!
!  All that is done for different connect speeds is to set the serial port
!  speed on the Macintosh to match the line speed.
!
@LABEL 11
note "Communicating at 1200 bps." 2
serreset 1200, 0, 8, 1
jump 16
!
@LABEL 12
note "Communicating at 2400 bps." 2
serreset 2400, 0, 8, 1
jump 16
!
@LABEL 13
note "Communicating at 4800 bps." 2
serreset 4800, 0, 8, 1
```

```
jump 16
!
@LABEL 14
note "Communicating at 9600 bps." 2
serreset 9600, 0, 8, 1
jump 16
!
@LABEL 15
note "Communicating at 19.2 kbps." 2
serreset 19200, 0, 8, 1
!
!  At this point the modems have connected.  If the script is answering a
!  telephone call, just exit right away and starting communicating.  If the
!  script is dialing out, give the other end some time (3 seconds in this
!  example) to get ready to talk to this modem.  Exit 0 tells Remote Access
!  that the script was successful in attempting a connection.
!
@LABEL 16
ifANSWER 17
pause 30
@LABEL 17
exit 0
!
!  Notice that the @ANSWER label is actually a comment here, and that
!  @ORIGINATE and @ANSWER start at the same place.  What's the point of having
!  separate entry points if they are not used?  Well, in the case of modems,
!  when they dial out or wait for a call, the setup is usually the same.  One
!  reason for separate entry points is when the script is not directly talking
!  to a modem, but maybe to a PBX or terminal server.  It may be necessary to
!  have completely different configuration for answering and originating
!  connections.
!
! @ANSWER
! Set up the modem to answer
!
@LABEL 30
write "ATS0=1\13"
matchstr 1 31 "OK\13\10"
matchread 30
jump 59
!
!  What is userhook 1 doing in label 32?  Here's the idea:  Either this script
!  controls a server that is waiting to answer the telephone, or it's waiting
!  for a callback to a connection that was initiated.  AppleTalk Remote Access
!  does a "passive" listen on the serial port (via the Serial Port Arbitrator)
!  so that other communications applications can use the serial port when ARA
!  is not using it.  When a call comes in for a server or callback, there
!  will be about 5-14 seconds while the modems negotiate the connection.
!  What would happen if a communications application on this Macintosh
!  wanted to use the serial port during that time?  Both connections
!  would fail.  The userhook 1 command tells ARA to mark the serial port in
!  use.  When that happens, applications that want to use the serial port will
!  be told it's busy, and the incoming connection can complete.  With that in
```

```
!  mind, the strategy below is:  When the modem receives a ring, jump to label
!  32, issue the userhook 1 command, then jump back up to label 31, wait for
!  the connect result code and continue processing the script.
!
@LABEL 31
matchstr 1  32 "RING\13\10"
matchstr 2  11 "CONNECT 1200\13\10"
matchstr 3  12 "CONNECT 2400\13\10"
matchstr 4  13 "CONNECT 4800\13\10"
matchstr 5  14 "CONNECT 9600\13\10"
matchstr 6  15 "CONNECT FAST\13\10"
matchstr 7  50 "NO CARRIER\13\10"
matchstr 8  50 "ERROR\13\10"
matchstr 9  52 "NO DIALTONE\13\10"
matchstr 10 53 "BUSY\13\10"
matchstr 11 54 "NO ANSWER\13\10"
matchread 700
jump 31
!
@LABEL 32
userhook 1
note "Answering phone..." 2
jump 31
!
!  These are some common error messages when the line is busy, no dialtone,
!  etc. They are documented in the Scripting Language Guide.  When the script
!  exits with a code other than zero, Remote Access knows that the connection
!  failed, and will inform the user with a dialog.
!
! 50: error messages
!
@LABEL 50
exit -6021
!
@LABEL 52
exit -6020
!
@LABEL 53
exit -6022
!
@LABEL 54
exit -6023
!
@LABEL 59
exit -6019
!
! Hang up the modem
! Note:  Why try to enter command mode and hang up the line with ATH, when
!        de-asserting DTR will always work, and it is used as a last resort
!        anyway?  If DTR is used immediately,  the modem will hang up
!        immediately.  This can have the ill effect of hanging up before all
!        the data in the modem's internal transmit buffer has been sent.
!        It is very desirable to have the last byte of data sent make
```

```
!        it out of the modem and across the phone line.  Typically,
!        the last packet sent is the disconnect packet, and if
!        the other side misses this packet, it may have to wait up to 45
!        seconds to hang up.
!
@HANGUP
@LABEL 60
settries 0
@LABEL 61
!
!  Here's the basic logic for hanging up: If the modem can be configured
!  to enter command mode when it receives a short break, send a short
!  break.  Send an ATH to hang the line up (and if possible up in the
!  configuration, set the modem to attempt to send all the data in the
!  buffer before it disconnects).  If that fails, it must still be on
!  line, so send the escape sequence to try to drop into command mode.
!  Don't issue a short break again since it did not work the first time.
!  If that fails, de-assert DTR which should force the modem to hang up
!  (make sure the cable is wired properly for this option!).
!  If +++ worked, don't send a short break again; flush the serial port
!  buffer in case the ATH failed due to any stray data hanging around.
!
!  How was this sequence determined?  Trial and error.  Different vendors'
!  modems behave differently when disconnecting.  Some modems will not enter
!  command modem during a disconnect, and the only option is to de-assert DTR
!  to force them to reset.  That's why DTR resets the modem instead of just
!  disconnecting it!  Experiment with this sequence to make it function, but it
!  should work with the majority of the modems available.
!
! Now, since the Telebit modems will drop into command mode when they receive
! a short break (S61=1), issue one here.  This will speed up the disconnect
! sequence by about 5-6 seconds.  Then continue on with normal AT disconnect
! processing.
!
Sbreak
!
! Wait just a brief amount of time (1/2 second in this case) so the modem will
! be ready to accept the ATH command.  Pause 1 actually seems to work ok, but
! it's set to 5 just to be safe.
!
pause 5
write "ATH\13"
matchclr
matchstr 1 63 "OK\13\10"
matchstr 2 63 "NO CARRIER\13\10"
matchstr 3 63 "ERROR\13\10"
matchread 30
inctries
iftries 3 63
! no response, try escape sequence
write "+++"
matchclr
matchstr 1 62 "OK\13\10"
```

```
matchread 15
!
! No response from modem, toggle DTR
!
DTRClear
pause 5
DTRSet
jump 61
! Pause 1 second to ensure we meet the escape time delay
@LABEL 62
pause 10
Flush
write "ATH\13"
matchstr 1 63 "OK\13\10"
matchstr 2 63 "NO CARRIER\13\10"
matchstr 3 63 "ERROR\13\10"
matchread 30
jump 61
!
!  Now that the modems have disconnected, and the script has possibly reset,the
!  modem, restore the factory settings.  Remember, the script may have hung up
!  the modem in order to get ready for a callback, or it wants to get ready to
!  wait to answer a call again.
!
! Recall factory settings
!
@LABEL 63
matchclr
matchstr 1 64 "OK\13\10"
write "AT&F0\13"
matchread 30
!
!  Now turn off auto answer if it was turned on to answer a call.  If this
!  script controls a server, the @ANSWER sequence will be called by ARA.
!  One other thing to watch out for here is that some modems expect to
!  talk to the DTE at the last connected speed.  If this is a V.32
!  modem and it just finished a connection with a 2400 baud modem, it
!  doesn't necessarily want to talk at 2400 the next time!  Some modems
!  don't exhibit this behavior, so play with it and see what happens.  Finally,
!  since it successfully hung up, exit the script with a result code of 0 to
!  let Remote Access know everything worked.
!
! Turn off auto answer, set S51 so modem will check interface speed on
! next command.  This is different than the Telebit T1600/T3000, which wants
! S51=255.
!
! S51=252 - Automatic speed selection, type ahead not permitted.
! S0=0    - Don't answer the phone if it rings.
!
@LABEL 64
matchclr
matchstr 1 65 "OK\13\10"
write "ATS51=252S0=0\13"
```

```
matchread 30
!
@LABEL 65
exit 0
```


Keywords: <None>

==========

This information is from the Apple Technical Information Library.

19960215 11:05:19.00

Tech Info Library Article Number: 10244