

AmiPhone

Jeremy Friesner

COLLABORATORS

	<i>TITLE :</i> AmiPhone		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Jeremy Friesner	February 12, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	AmiPhone	1
1.1	Contents	1
1.2	Installation	1
1.3	credits	2
1.4	How to reach me	3
1.5	disclaimer	3
1.6	voicemail	3
1.7	AmiPhone is DonationWare	4
1.8	requirements	5
1.9	introduction	6
1.10	Using AmiPhoned	6
1.11	amiphonedgui	6
1.12	amiphonedmenus	7
1.13	AmiPhoned's Project Menu	7
1.14	AmiPhoned's Relay Menu	8
1.15	relays	8
1.16	Relay Methods	9
1.17	Using AmiPhone	11
1.18	Environmental variables	11
1.19	receiving	11
1.20	keys	12
1.21	micstates	12
1.22	inputvolumeindicator	13
1.23	The Samples per Second slider	13
1.24	The Bandwidth meter	14
1.25	The AmiPhone GUI	14
1.26	xmitdelayslider	15
1.27	threshvolslider	15
1.28	Starting AmiPhone	15
1.29	ToolType read by AmiPhoned	17

1.30 idleratearg	17
1.31 tcpbatchxmitarg	17
1.32 micgainarg	18
1.33 awayvararg	18
1.34 maxvoicemailsizearg	19
1.35 maxmessagesizearg	19
1.36 daemonleftarg	19
1.37 daemontoparg	19
1.38 showdaemonarg	20
1.39 voicemaildirarg	20
1.40 inputchannelarg	20
1.41 samplerarg	21
1.42 invertwaveformarg	21
1.43 phonebookarg	22
1.44 exptooltypes	22
1.45 maxsampleratearg	24
1.46 maxxmitdelayarg	24
1.47 techniquearg	25
1.48 inputsourcearg	25
1.49 amplifyarg	25
1.50 holdtotransmitarg	26
1.51 sendpriarg	26
1.52 receivepriarg	26
1.53 maxbandwidtharg	26
1.54 sampleratearg	27
1.55 xmitonplayarg	27
1.56 enableonconnectarg	27
1.57 Specify how often to transmit a packet	27
1.58 Specify the minimum volume (threshold)	28
1.59 Specify a compression algorithm to use	28
1.60 Specify a computer to connect to on startup	28
1.61 Specify a Public Screen	28
1.62 Specify window X-coordinate	29
1.63 Specify window Y-coordinate	29
1.64 AmiPhone menus	29
1.65 messagesmenu	29
1.66 Project Menu	30
1.67 TCP Menu	30
1.68 messagebrowserwindow	30

1.69	Settings Menu	31
1.70	Sound Transmission technique and limitations	32
1.71	phoneutil	33
1.72	digitalamplification	33
1.73	situation	34
1.74	Thanks	34
1.75	faq	34
1.76	The ground was littered with squashed bugs...	36
1.77	How can I check that everything is okay?	43
1.78	What's Next?	44
1.79	Known Bugs and Other Problems	44
1.80	otherprogs	44

Chapter 1

AmiPhone

1.1 Contents

AmiPhone V1.80 by **Jeremy Friesner**

AmiPhone is a program that allows you to have a voice conversation with a friend over the Internet, using your computer's audio output, and an 8 bit parallel port digitizer and microphone.

NOTE: AmiPhone 1.80 is not compatible with AmiPhone 1.41β or lower!

Make sure anyone you connect to has the latest version installed!

Disclaimer Don't blame me!

Distribution AmiPhone is DonationWare!

Requirements What do I need to run this program?

Introduction What does AmiPhone do?

Installation How do I set AmiPhone up?

Using AmiPhone How to run the AmiPhone client

Using AmiPhoned How to use the AmiPhoned server

Using PhoneUtil How to use this dorky little utility

Credits Where it's due

Acknowledgments Thanks to...

History Bug fixes and enhancements

Future What next?

F.A.Q. Frequently Asked Questions

Known Problems Bugs! Aack!

Other programs Plug, plug!

1.2 Installation

To install AmiPhone, either run the supplied Installer script, or follow the directions below.

To install AmiPhone manually:

- 1) Set the **SAMPLER ToolType** of the AmiPhone icon to the sampler you are going to use. If you have a parallel port digitizer which is not explicitly supported, set this ToolType to "GENERIC"--it may still work.
- 2) If you wish to use voice mail, set at least the **VOICEMAILDIR** and **AWAYVAR** ToolTypes in the AmiPhone icon appropriately. You also may wish to set the **MAXMESSAGE SIZE** and **MAXVOICEMAILSIZE** ToolTypes, to limit disk space used by callers.
- 3) Copy the files AmiPhone and AmiPhone.info to where you wish to keep them. AmiTCP:bin is the recommended location; if you keep AmiPhone there, you can skip to step 5.
- 4) If you are keeping the AmiPhone executable in a directory other than AmiTCP:bin, you also need to set the ENV variable AMIPHONE to the fully qualified path and filename of the AmiPhone executable. e.g. if you keep the AmiPhone executable in sys:utilities, you would do this:
setenv AMIPHONE sys:utilities/AmiPhone
copy env:AMIPHONE envarc:
- 5) Copy the file AmiPhoned to amitcp:serv.
- 6) Add the following line to the end of your amitcp:db/services file:
AmiPhone 2956/tcp
- 7) Add the following line to the end of your amitcp:db/inetd.conf file:
AmiPhone stream tcp nowait root amitcp:serv/AmiPhoned
- 8) If you don't have the env variable HOSTNAME set on your computer, set it to reflect your Amiga's Internet host name. (e.g. mine is jfriesne.extern)
- 9) Re-start your computer (just to be sure of things), and try it out!
It's probably best to start by connecting to localhost, just to make sure your digitizer is working. Once you've got that working (you can speak and hear your own voice on your speakers), connect to someone else.

1.3 credits

AmiPhone V1.80

Created by **Jeremy Friesner**

ADPCM compression routines by Christian Buchner

Compiled with DICE C by Matt Dillon

Interrupt handlers assembled with PhxAss by Frank Wille

1.4 How to reach me

Here are some ways to get in touch with me:

by EMail: jfriesne@ucsd.edu

(note: might be invalid after June '96!)

by SMail: Jeremy Friesner

4680 Mt. Longs Drive

San Diego, CA 92117

1.5 disclaimer

This software comes with no warranty, either expressed or implied.

The **author** is in no way responsible for any damage or loss that may occur due to direct or indirect usage of this software. Use this software entirely at your own risk.

1.6 voicemail

AmiPhone can be used to send and receive voice mail messages over the Internet.

In order to receive a voice mail message, your Amiga must be on a "live" connection to the Internet. (i.e. AmiPhone voice mail is not based on a store-and-forward methodology like e-mail, but rather it works like an answering machine; it will "pick up the receiver", listen and record incoming audio for you.)

If someone tries to connect to your Amiga with AmiPhone and you're **not there**, AmiPhoned will give them the option of leaving a message.

If they choose to do so, AmiPhoned will save the message in a file in the **VOICEMAILDIR**, and you will be able to listen to it later.

In order for voice mail to work, AmiPhone must have several Tool Types set correctly. The first, **VOICEMAILDIR**, tells AmiPhoned which directory to save message files in. The second, **AWAYVAR** tells AmiPhone the name of an env: variable to look for. If AmiPhoned sees that that variable is set, it will assume you're away from your computer, and offer to take a message. If that variable is not set, it will put up a connection requester, as before. There are also some optional Tool Types controlling message sizes, such as **MAXMESSAGE SIZE** and **MAXVOICEMAILSIZE**.

To check your messages, run AmiPhone and select "Messages..." from the **Messages Menu** . This will bring up the **Message Browser Window** . From there you will be able to play any messages you have.

Testing

It's a good idea to test out your voice mail reception before having other people try it. To do this, just set your **AWAYVAR** to 1 or whatever, (e.g. go to a shell, and do a "setenv BLANKED 1") and then connect to "localhost". You should see an EasyRequester asking you if you want to leave a message. Click on "Leave Message", then say hello (or whatever) to yourself, and disconnect. Then select **Messages...** from the messages menu and play back the sound file that is there. You should hear your message played back.

1.7 AmiPhone is DonationWare

NOTE: Please **report** any bugs you find while using this software.

AmiPhone may be distributed freely, as long as the original archive is kept intact.

AmiPhone is DonationWare. I've put a lot of time into it to make it as fun and useful as possible, so if you find AmiPhone to your liking and use it often, please consider sending **me** a \$5 or \$10 donation, and in return I will send you the source code to AmiPhone (if you want it), future upgrades directly and give your suggestions preferred treatment. However, if you can't afford that or for some other reason don't want to send money, that's okay also. Just send me email telling me that you're using it, and list any suggestions that you have for improving it. :-)

Permission is given to include this program in a public archive (such as a BBS, FTP site or PD library) providing that all parts of the original distribution are kept intact. These are as follows:

Listing of archive 'AmiPhone1.80.lha':

Original Packed Ratio Date Time Name

```
-----
2273 556 75.5% 11-May-96 23:13:22 AmiPhone.info
67448 34776 48.4% 11-May-96 23:13:18 +AmiPhone
93344 33642 63.9% 11-May-96 23:13:20 +AmiPhone.guide
4274 1244 70.8% 11-May-96 23:13:22 +AmiPhone.guide.info
3879 2364 39.0% 11-May-96 23:13:18 +AmiPhone.info
29800 16742 43.8% 11-May-96 23:13:20 +AmiPhoned
35324 19343 45.2% 11-May-96 23:13:20 +AmiPhoned_debug
```

4123 1604 61.0% 11-May-96 23:13:22 +EditTextFile.rexx
 10279 3437 66.5% 11-May-96 23:13:20 +Install_AmiPhone
 1644 1252 23.8% 11-May-96 23:13:20 +Install_AmiPhone.info
 10776 6594 38.8% 11-May-96 23:13:22 +PhoneUtil
 2487 1190 52.1% 11-May-96 23:13:22 +README
 1096 485 55.7% 11-May-96 23:13:22 +README.info

266747 123229 53.8% 11-May-96 23:13:38 13 files

No charge may be made for this program, other than a reasonable copying fee, and/or the price of the media.

1.8 requirements

AmiPhone requires an Amiga running Kickstart V37 (WorkBench 2.04) or higher to operate.

AmiPhone also requires AmiTCP3.0b or higher, and either a Toccata sound board, an Aura PCMCIA card, or an 8-bit parallel-port sampler. (at least, if you want to transmit any sound--you can receive sound without this).

Furthermore, to make any decent use of AmiPhone, you need a connection to a network that goes at least 14.4 kilobits per second. The faster the connection, the better!

Lastly, a fast CPU is recommended. Sound sampling is a CPU-intensive process, and slow Amigas will be confined to transmitting at very low sampling rates.

AmiPhone opens the following libraries and devices:

Library Min Version # Library location/Comment

intuition.library 37 in ROM

bsdsocket.library 2 e.g. AmiTCP must be running

graphics.library 37 in ROM

gadtools.library 36 in ROM

icon.library 33 in ROM

asl.library 37 in LIBS:, required to use File Requesters

toccata.library 6 in LIBS:, required to use Toccata digitizer

timer.device ? in ROM

ahi.device ? in DEVS:, required to use AHI

1.9 introduction

Please read the [History](#) section for information on changes and bug-fixes.

AmiPhone is a program that simulates a telephone conversation over an Internet connection. The main advantage to this is that you can talk to any other AmiPhone user on the Internet for the cost of a call to your Internet Service Provider (i.e. local). The disadvantage is that audio requires a [good deal of throughput](#) , and that it is difficult to shoehorn a realtime audio stream into the packet-based world of TCP/IP networks. AmiPhone operates by getting audio data from the digitizer (you must have a Toccata board or an 8-bit sampler hooked up to the parallel port to send sound with AmiPhone) and sending it as a series of UDP packets to the AmiPhone daemon at the other end of the connection, where the data is played out through the other person's speakers. When the other person talks, the same thing happens in reverse. As of version 0.8β, AmiPhone can also be used to send and receive voice mail. Read the section on [voice mail](#) to learn how.

As of version 1.1β, AmiPhone can also be used to broadcast sound from one source to many destinations. Read the section of [relays](#) to learn how.

1.10 Using AmiPhoned

AmiPhoned is the "server" portion of the AmiPhone setup. Its job is to play any incoming sound, and possibly forward that sound to another AmiPhoned. There will be one AmiPhoned incarnation running for each incoming connection you accept.

[The GUI](#) AmiPhoned has a GUI?

[Startup options](#) More command line arguments & ToolTypes

[Env variables](#) Environmental variables that AmiPhoned uses

[Menu options](#) Explanation of AmiPhoned's menus

[Relays](#) Relay intro/tutorial

1.11 amiphonedgui

Yes, it's true. AmiPhoned now has a simple GUI of its own. The AmiPhoned GUI consists of a title bar and some attached menus. The title bar reads "AmiPhoned:" and the name of the computer it is

receiving sound from.

On the right edge of the title bar is an asterisk that flashes whenever data is being received. If there is an error in the received data, or a problem playing the sound (e.g. AmiPhoned couldn't allocate a sound channel, or a packet was skipped), then the asterisk will turn into a pulsating 'X'.

This GUI is not always visible, however. Unless you have set the **SHOWDAEMON** ToolType to do otherwise, the AmiPhoned GUI will only appear when you have initiated a receive-only AmiPhoned session.

Otherwise, it is assumed that you can access the AmiPhoned GUI via AmiPhone client window if you want it, and thus the AmiPhoned window will not automatically appear.

If you are running an AmiPhone client window, you may make the associated AmiPhoned GUI appear and disappear by selecting the "Show Daemon" item in the AmiPhone **TCP Menu**. Otherwise, the only way to make the AmiPhoned window appear is to send a CTRL-E signal to its process using "break" or some other utility. Once the AmiPhoned GUI is visible, you may hide it by selecting "Hide" from its **Project Menu**.

Clicking on the close bar of the AmiPhoned GUI is identical to selecting "Quit" from the **Project Menu**. It will close the AmiPhone connection and cause AmiPhoned to go away.

The AmiPhoned GUI is the only way to create **relay connections**.

1.12 amiphonedmenus

AmiPhoned currently has two menus. To access them, the AmiPhoned title bar must be the current window.

Project Menu Boring stuff

Relay Menu To add/remove relays

1.13 AmiPhoned's Project Menu

Record - This item is a toggle. If it is checked, any sound AmiPhone receives will be stored into the voice mail directory. You may turn recording on or off at any time. Only one message file will be created per AmiPhoned session, and it will contain the concatenation of all sound received while this item was checked.

If voice mail has not been set up, then this option will not

be available.

Hide - Selecting this option makes the AmiPhoned GUI invisible.

Click [here](#) for ways to make it visible again.

About - Displays an EasyRequester giving version numbers, etc.

Quit - Closes the AmiPhone connection and exits, stage left.

1.14 AmiPhoned's Relay Menu

The AmiPhoned Relay menu is what allows you to create and remove **relays** . There are ten items in this menu. Initially, they all say "Add Relay #x". (x being a number between 0 and 9)

Selecting one of these items will cause a string requester to pop up and ask you for an IP name of a computer to relay your incoming sound to.

Once you have entered a name, that name (or at least part of it, as space allows) will appear in that menu entry, in parentheses. The parentheses mean that the connection has been started, but has not yet been fully established.

If the person at the far end agrees to accept the relay, the parentheses are removed from the menu item. After this occurs, the relay is fully operational, and any sound data that comes to you will be passed along to the relay machine.

To disconnect from a relay (i.e. stop relaying data to it), just select its name from the menu. The menu item will revert to "Add Relay #x", and the connection will be closed.

1.15 relays

Relays are a neat new ability of AmiPhoned which allow audio to be broadcast over an arbitrary number of networked Amigas, instead of just between two conversants. The idea is that any AmiPhoned server, in addition to playing the data packets that it receives, can also forward these packets on to up to 10 other Amigas. In practice, however, most Internet connections will not have bandwidth for more than one or two relays, but even one relay per machine can allow broadcasting to an indefinite number of Amigas.

To use relays, you must bring up and use the **AmiPhoned GUI** , and in particular, its **Relay** menu.

If the computer feeding your AmiPhoned disconnects from you, your AmiPhoned will also disconnect from all of the AmiPhoned's it is forwarding the data to.

Click [here](#) for info on relay forwarding configurations.

1.16 Relay Methods

The strain on a given Internet connection will increase directly (and often very quickly) as the number of relays emanating from that connection grows. With that in mind, here are some ways to set up a broadcast, complete with nifty ASCII charts. :)

1) The "AmiPhoned chain"

AmiPhone -> AmiPhoned -> AmiPhoned -> ... -> AmiPhoned

In this method, each AmiPhoned has only one relay, so that all the Amigas are linked in a single chain.

Advantages: Every Amiga needs only to forward one packet out for each packet received. No one will bear any additional load as the number of receivers increases.

Disadvantages: A single weak point in the chain will affect everyone after it.

2) The "AmiPhoned tree"

/-> AmiPhoned -> ...

/

/

/

/-> AmiPhoned ->

/\

/\

/\

/\-> AmiPhoned -> ...

AmiPhone -> AmiPhoned ->

\/-> AmiPhoned -> ...

\/

\/

\/

\-> AmiPhoned ->

\

\

\

\-> AmiPhoned -> ...

In this method, each AmiPhoned has one or more relays, so that the broadcast spreads out in a tree-like fashion.

Advantages: The load on each Amiga can still be relatively low, but breakdowns are somewhat less likely to wipe out

the whole group. It's also more flexible, in that you don't have to figure out who is last in the list--anyone with spare capacity can add a relay.

Disadvantages: Two or more relays may be too much for slow links, especially modem-based Amigas.

3) The "AmiPhoned broadcast"

l-> AmiPhoned

|

|

l-> AmiPhoned

|

|

l-> AmiPhoned

|

|

l-> AmiPhoned

|

|

AmiPhone -> AmiPhoned -l-> AmiPhoned

|

|

l-> AmiPhoned

|

|

l-> AmiPhoned

|

|

l-> AmiPhoned

|

|

l-> AmiPhoned

In this configuration, one central AmiPhoned relays packets to all of the others.

Advantages: No relay duty required for any of the receiving AmiPhoned's.

Disadvantages: The distributing AmiPhoned better be running on a heavy-duty connection! Also, the broadcast is limited to ten receivers, tops.

4) Some combination of the above. This is no doubt the kind of configuration that will actually be used. If anyone has any opinions on what works best, I'd be happy to **hear them** . :)

1.17 Using AmiPhone

The GUI Where do I click?

Startup options Command line arguments & ToolTypes

Env variables Environmental variables that AmiPhone uses

Menu options Explanation of cryptic menus

Keyboard Shortcuts For people too cool to use the mouse

Receiving a call How to cope with popularity

Voice mail Voice mail intro/tutorial

1.18 Environmental variables

There are currently three ENV: variables that AmiPhone uses.

1) If you wish to keep AmiPhone in a directory other than amitcp:bin, you should set the ENV variable AMIPHONE to the fully qualified pathname of the AmiPhone executable. This will help AmiPhoned find AmiPhone in order to launch it. If AMIPHONE is not set, AmiPhoned will default to running "amitcp:bin/AmiPhone".

2) The ENV variable HOSTNAME must be set to reflect your Amiga's Host Name on the Internet. Without this, AmiPhoned will not be able to accept connections.

3) The third ENV variable is used by AmiPhoned to determine whether you are at the computer or not. If this ENV variable is present, and you have voice mail set up, AmiPhoned will take a message instead of putting up a requester. This ENV variable's name is user-selectable through the **AWAYVAR** ToolType.

Note that the Installer script will set these for you if necessary.

1.19 receiving

When somebody on the Internet tries to call you with AmiPhone, an EasyRequester will appear on your Workbench screen indicating who is trying to call, and giving you some options, which may include:

1) Receive and Transmit - This will allow the incoming audio and launch a local AmiPhone client to let you transmit data

back to the caller.

2) Receive only - This will allow the caller to send audio to you, but you will not be able to send audio to them.

3) Take a message - This will only appear if you have voice mail

set up properly, and your voice mailbox isn't full. Selecting this will cause AmiPhoned to invite the caller to leave a message, which AmiPhoned will save to your **VOICEMAILDIR** directory. You will be able to hear the message as it is recorded.

4) Deny - This will turn down the caller's request for an AmiPhone session, and close the connection.

1.20 keys

There are keyboard equivalents in AmiPhone for most options available in the menus; those items with keyboard equivalents have the equivalent listed next to the item.

Furthermore, there are also these keyboard shortcuts:

(space bar) : Same effect as clicking on the **Microphone button**

D : Disables sampling at any time.

E : Enables sampling at any time sampling is possible.

, : Reduces the sampling rate by 10 bytes per second.

< : Reduces the sampling rate by 50 bytes per second.

. : Increases the sampling rate by 10 bytes per second.

> : Increases the sampling rate by 50 bytes per second.

1.21 micstates

Possible states of the Microphone button, and their meanings:

QUIET: This means that AmiPhone is sampling the parallel port, ready to send data, but that the sound coming in from the port isn't louder than the minimum volume threshold, and thus is not being transmitted. You can adjust this threshold with the **Thresh Vol** slider or set the default via the **THRESHVOLUME** startup option. Clicking on the microphone button while it is in this state will cause it to become **DISABLED**.

XMITTING: This means that the sound coming from the digitizer is being sent to the remote Amiga. Clicking on the microphone while it is in this state will cause it to become **DISABLED**.

note: If you play back an AmiPhone message from disk while the **Xmit on Play** option is checked, you will see a picture of an audio tape in the button. This means that the sound being played back is also being transmitted.

NOCONN: This means that the sampler is not enabled, because you are not connected to another Amiga.

DISABLED: This means that you are connected, but that the sampler is not active. Clicking on the microphone button starts the sampler, and puts the Microphone button in either the QUIET or XMITTING state (depending on input line volume).

1.22 inputvolumeindicator

The Input Volume Indicator is a continuously updated vertical bar that shows the volume of input being received from your microphone. If the bar is completely full, then the input volume is **probably** too loud (i.e. it is being distorted). If the bar is empty, then little or no sound is coming through the microphone.

There is a little horizontal black line that crosses this indicator, usually near the bottom. That line represents the volume threshold underneath which AmiPhone will not transmit the sound. You can change the volume threshold via the **THRESHVOLUME** startup option or the **Thresh Vol** slider.

This bar does not function when the microphone is disabled.

1.23 The Samples per Second slider

The Samples per Second slider allows you to choose the sampling rate used to create and transmit samples. (Note: it has no effect on the sampling rate of samples received--that is up to your partner)

You may set this slider to suit your setup, and you may set the slider's default value by using the **SAMPLERATE** startup option.

Benefits of a high samples per second setting:

- better sounding samples (you don't lose the high frequencies)

Benefits of a low samples per second setting:

- less bandwidth used on your network connection
- less CPU used (because there is less data to compress and decompress, and because there are fewer sampling interrupts to process)

Sampling speeds range from 1600 to 9999 samples per second, although you can try sampling at even higher rates by specifying the **MAXSAMPLERATE** startup argument.

1.24 The Bandwidth meter

The bandwidth meter is a scrolling graph that shows how many bytes per second are being sent and received by AmiPhone.

The graph represents time on the horizontal axis, and bytes per second on the vertical axis. The bottom of the graph represents zero bytes per second, and the top represents **MAXBANDWIDTH** bytes per second.

If **MAXBANDWIDTH** is not set, the default maximum graph value visible on the graph is 2,880 bytes per second.

At every 1,440 bytes per second interval along the graph, a dashed reference line is plotted.

The colored areas in the graph denote how much bandwidth is being used for receiving data (the lower color), and for sending data (the upper color).

The height of these colors is affected by your **sampling rate** and your choice of **compression algorithm** . If you choose a sampling rate/compression algorithm combination such that the colored bars don't rise higher than the bandwidth of your Internet connection, you will not overload the connection.

As this graph demonstrates, you can select higher sampling rates as long as you and your partner don't talk at once.

If one of the colors in the graph turns into a different color (usually just for a line or two, although it could be more if your connection is broken, too slow, or too flaky for your settings), this means that AmiPhone is having problems with the corresponding phase of the transmission. i.e. if the upper color changes, you are having problems sending data, or if the lower color changes, you are having problems receiving data. If this happens a lot, try a lower sampling rate, or more compact compression algorithm.

1.25 The AmiPhone GUI

Besides the **Menus** , there are several parts to the AmiPhone GUI.

The first is the microphone (transmission) button. This button acts as an indicator for what state the transmission device is in, as well as a way to change its state. For a description of the possible states, click [here](#) .

Adjacent to the microphone, on its right, is the **input volume indicator** .

It shows how much sound is currently coming through the digitizer.

In the middle of the window are three horizontal slider bars:

The **top slider** controls the sampling frequency of your digitizer.

The **middle slider** controls how much data is sent at once.

The **bottom slider** controls the minimum sample volume necessary to trigger the transmission of the sample.

The scrolling graph on the right side of the window is a **meter** of the amount of bandwidth you are using.

1.26 xmitdelayslider

This slider lets you control how many milliseconds of sound AmiPhone will sample before it sends the data it has collected to your peer's computer. The delay can range from 90 to 700 milliseconds. Smaller values give you better response time, but can lead to clipping of the ends of your sentences and they place some extra strain on your system.

The default value for this slider can be set via the **XMITDELAY** ToolType.

1.27 threshvolslider

This slider lets you control how sensitive the trigger is for AmiPhone's silence suppression filter. The slider ranges from 0 to 130. At higher values, it takes more noise to cause AmiPhone to send data. Thus, at 130 AmiPhone will never send data, whereas at 0 AmiPhone will always send data.

The default value for this slider can be set via the **THRESHVOLUME** ToolType.

1.28 Starting AmiPhone

AmiPhone can be started from either the Shell or the WorkBench. It supports quite a few command line arguments. Each of these arguments also has a ToolType equivalent. Look in the AmiPhone icon for examples of these ToolTypes--they are there, just commented out with parentheses.

AmiPhone will try to find parameters in its icon, even if you started it from the shell. Command line arguments override icon ToolTypes, however.

Template: AmiPhone PEERNAME/A, TOP/K/N, LEFT/K/N, COMPRESS/K, PUBSCREEN/K, CONNECT/K, THRESHVOLUME/K/N, XMITDELAY/K/N, SAMPLERATE/K/N, MAXBANDWIDTH/K/N, SENDPRI/K/N, RECEIVEPRI/K/N, MAXSAMPLERATE/K/N, MAXXMITDELAY/K/N, SAMPLETECHNIQUE/K, SAMPLER/K, INPUTCHANNEL/K,

VOICEMAILDIR/K, AMPLIFY/K/N, INPUTSOURCE/K, PHONEBOOKx/K,
MICGAIN/K/N, IDLERATE/K/N, ENABLEONCONNECT/S,
XMITONPLAY/S, HOLDTOTRANSMIT/S, INVERTWAVEFORM/S,
BATHTCPXMIT/S

PEERNAME/A Connects to a computer on startup

TOP/K/N Assigns default Y co-ordinate to window

LEFT/K/N Assigns default X co-ordinate to window

PUBSCREEN/K Opens AmiPhone on given Public Screen

CONNECT/K Connects to a computer on startup

COMPRESS/K Specifies a default compression algorithm

THRESHVOLUME/K Specifies a the trigger volume for transmission

XMITDELAY/K Specifies the granularity of sampling

SAMPLERATE/K Specifies the default sampling rate

MAXBANDWIDTH/K Specify the top of the bandwidth graph

SENDPRI/K Specify the task priority of AmiPhone

RECEIVEPRI/K Specify the task priority of AmiPhoned

SAMPLETECHNIQUE/K Use hard or soft interrupts?

MAXSAMPLERATE/K Set the maximum possible sampling rate

MAXXMITDELAY/K Set the maximum possible packet interval

SAMPLER/K Inform AmiPhone as to which sampler you're using

INPUTCHANNEL/K Select a default channel to sample from

VOICEMAILDIR/K Specify a directory to store voice messages in

AMPLIFY/K/N Specify a digital signal amplification value

INPUTSOURCE/K Specify an input source

PHONEBOOKx/K To put an entry in the "Connect To" submenu

MICGAIN/K/N Specify a default analog input amplification

IDLERATE/K/N Specify the sampling rate during silent periods

ENABLEONCONNECT/S Enable the microphone on connect?

XMITONPLAY/S Transmit messages you are replaying?

HOLDTOTRANSMIT/S Transmit only when you hold down the button

INVERTWAVEFORM/S Use if the silence detection isn't working

BATHTCPXMIT/S Send whole sentences at once over slow links

There are also a few ToolTypes that are read by **AmiPhoned** .

Make sure to read about these, some are important!

Finally, there are some ToolTypes that can be used for experimentation digitizer settings. Click [here](#) to read more about these.

1.29 ToolType read by AmiPhoned

Since AmiPhoned cannot be launched from the CLI, these ToolTypes must be specified via AmiPhone's icon. AmiPhoned will scan the icon of the file specified in the ENV: variable **AMIPHONE**, and if that fails, (because AMIPHONE isn't set or isn't valid), it will look at `amitcp:bin/AmiPhone.info`. AmiPhoned will NOT look for an AmiPhoned icon, so don't try to create one for it.

ToolTypes that AmiPhoned currently looks for:

SHOWDAEMON/K Tell AmiPhoned whether or not to show info bar

DAEMONLEFT/K Default X co-ordinate to AmiPhoned info bar

DAEMONTOP/K Default Y co-ordinate to AmiPhoned info bar

PUBSCREEN/K What Public Screen the info bar should use

MAXVOICEMAILSIZE/K Max disk space to use for messages

MAXMESSAGE/K Max disk space to use for each message

AWAYVAR/K ENV var to check to determine user presence

1.30 idleratearg

This startup argument allows you to set the rate at which AmiPhone will sample the parallel port during "idle" periods-- i.e. when there is no sound detected from the sampler. The default idle sampling rate is 200Hz, but if you wish you may vary the rate from 11Hz to 32767Hz. (Of course, making your idle rate higher than your sampling rate strikes me as a bit silly, but then who am I to judge? ;^))

Usage Example:

AmiPhone IDLERATE=900

1.31 tcpbatchxmitarg

This startup argument allows you to start AmiPhone with the **TCP Batch Xmit** option in the settings menu already checked.

Usage Example:

AmiPhone TCPBATCHXMIT

1.32 micgainarg

This startup argument allows you to set AmiPhone to use a given hardware amplification level for the audio being digitized. The way in which this amplification is achieved varies from digitizer to digitizer, so the arguments to this ToolType vary slightly as well.

This startup argument affects the following digitizers:

DSS8 - a value between zero (no amplification) and 7 (highest amplification) can be selected.

Example: AmiPhone SAMPLER=DSS8 MICGAIN=0 [no gain]

AmiPhone SAMPLER=DSS8 MICGAIN=3 [medium gain]

AmiPhone SAMPLER=DSS8 MICGAIN=7 [highest gain]

TOCCATA - the toccata only supports two levels of gain; +0dB, and +20dB.

Example: AmiPhone SAMPLER=TOCCATA MICGAIN=0 [+0dB]

AmiPhone SAMPLER=TOCCATA MICGAIN=1 [+20dB]

PERFECTSOUND - This digitizer has a way to change the current amplification by a given amount, but no known way to read the current setting! Thus MICGAIN may not give predictable results with this sampler.

Example: AmiPhone SAMPLER=PERFECT MICGAIN=3 [raise level by 3]

AmiPhone SAMPLER=PERFECT MICGAIN=-3 [lower level by 3]

1.33 awayvararg

This ToolType is used specify an ENV variable that AmiPhoned will look at whenever an incoming call arrives. If the ENV var specified by this ToolType exists, AmiPhoned will assume the user is not around to receive the call, and offer to take a message instead.

If this ToolType is not set, AmiPhoned will not take any messages.

THIS ARGUMENT CAN ONLY BE SPECIFIED AS A TOOL TYPE IN THE AMIPHONE ICON!

Usage Example:

AWAYVAR=BLANKED

(Many screenblankers set the variable BLANKED whenever they are active. This example would cause AmiPhoned to take messages whenever the screen blanker is on)

1.34 maxvoicemailsizearg

This ToolType can be used to specify the total number of kilobytes that can be used at any one time by AmiPhoned to store messages. AmiPhoned will not let the size of the voice mail directory become larger than this number of kilobytes.

THIS ARGUMENT CAN ONLY BE SPECIFIED AS A TOOL TYPE IN THE AMIPHONE ICON!

Usage Example:

```
MAXVOICEMAILSIZE=500
```

(Sets a 500 kbyte limit on the voice mail directory)

1.35 maxmessagesizearg

This ToolType can be used to specify the maximum number of kilobytes that a single message can be. When an incoming message file reaches this limit, AmiPhoned will hang up the connection.

THIS ARGUMENT CAN ONLY BE SPECIFIED AS A TOOL TYPE IN THE AMIPHONE ICON!

Usage Example:

```
MAXMESSAGE SIZE=100
```

(Sets a 100 kbyte limit on the size of individual incoming messages)

1.36 daemonleftarg

This ToolType can be used to specify the horizontal coordinate of the AmiPhoned info bar. For example, if you wish the info bar to appear 100 lines from the left of the screen, set a ToolType in AmiPhone's icon to

```
DAEMONLEFT=100
```

THIS ARGUMENT CAN ONLY BE SPECIFIED AS A TOOL TYPE IN THE AMIPHONE ICON!

1.37 daemontoparg

This ToolType can be used to specify the vertical coordinate of the AmiPhoned info bar. For example, if you wish the info bar to appear 200 lines from the top of the screen, set a ToolType in AmiPhone's icon to

```
DAEMONTOP=200
```

THIS ARGUMENT CAN ONLY BE SPECIFIED AS A TOOL TYPE IN THE AMIPHONE ICON!

1.38 showdaemonarg

This ToolType can be used to tell AmiPhoned when it is appropriate to display its info bar. The info bar is useful when you are using **Listen Only** mode, because it gives you an easy way to close the connection (by clicking its close box). Thus the default behavior (when this ToolType is not specified) is to only display the info bar when you have selected "Listen Only" from an incoming connection request.

However, some people may not like having the info bar in any case, and others may want it all the time. So, setting

```
SHOWDAEMON=YES
```

in the AmiPhone icon's ToolTypes will cause the info bar to always appear when AmiPhoned is active, and setting

```
SHOWDAEMON=NO
```

will cause the info bar to never appear. (This can be a pain in the neck if you are using relays)

THIS ARGUMENT CAN ONLY BE SPECIFIED AS A TOOL TYPE IN THE AMIPHONE ICON!

1.39 voicemaildirarg

This Tool Type tells AmiPhone and AmiPhoned what directory to use to store and retrieve voice mail. If it is not set as a Tool Type, AmiPhoned will not attempt to save incoming messages. You can specify this on the command line to AmiPhone, for the purpose of hearing saved messages; but AmiPhoned will not accept voice mail unless it is in the Tool Type list in the AmiPhone icon.

It is important that your incoming messages have their own directory (i.e. no other files should be in that directory!)

Example:

```
VOICEMAILDIR=work:Messages/incoming
```

1.40 inputchannelarg

With this argument, you can set the default channel AmiPhone will sample from. If this argument is not specified, AmiPhone will sample from the left input channel. Note that some digitizer types may not support this.

Usage Example:

```
AmiPhone INPUTCHANNEL=RIGHT
```

1.41 samplerarg

With this argument, you can let AmiPhone know which sampler you're using. Many samplers will work with GENERIC (the default), but if you set this argument to your sampler type, AmiPhone may have more features enabled and/or work better for you.

Valid sampler keywords currently are:

GENERIC

This is the default setting. AmiPhone assumes no special features on your digitizer.

DSS8

Set this when using the GVP DSS8 or DSS8+ digitizers.

PERFECTSOUND

Set this when using the PerfectSound series digitizers.

TOCATA

Set this to use the Tocatta Zorro II sound card as a (8-bit) digitizer.

AURA

Set this to use the Aura PCMCIA card as a (8-bit) digitizer.

AMAS

Set this when using the A.M.A.S. digitizer

SOUNDMAGIC

Set this when using the Sound Magic digitizer

CUSTOM

This setting allows you to program the digitizer by hand, using certain **ToolTypes** .

Usage Example:

```
AmiPhone SAMPLER=PERFECTSOUND
```

1.42 invertwaveformarg

With this startup argument, you can cause AmiPhone to read the sampler's waveform in an "inverted" fashion, so that silence is read as zero, and full volume as 255. (As for why this is considered the "inverted" way... don't ask ;)) Why should you care about this? Because with some samplers, silence is read as full volume, and full volume is read as silence, causing the automagic silence detection mechanism to work backwards.

If you are experiencing that problem (i.e. every time you enable

the sampler it starts transmitting, even if you're not saying anything), try this startup argument and see if that fixes it. Whether it does or not, please email [me](#) and let me know what your [situation](#) is, so that I can further my efforts to fix this type of problem.

This argument will have no effect when used with the PerfectSound or Toccata sampler types.

Usage Example:

```
AmiPhone INVERTWAVEFORM
```

1.43 phonebookarg

This ToolType is actually ten separate ToolTypes. They are the sole way to configure the "Connect To" submenu in the [TCP menu](#) .

The ToolTypes are named PHONEBOOK1, PHONEBOOK2, ..., PHONEBOOK9, PHONEBOOK0. Their argument should be of the format:

```
<name to be seen in menu>:<IP address>
```

Note that quotes are necessary if this parameter is used from the CLI and there are spaces in the arguments. Quotes are never necessary when specifying these options via ToolTypes.

Usage Example:

```
AmiPhone "PHONEBOOK0=Loopback Device:localhost"
"PHONEBOOK1=Jeremy Friesner:jfriesne.extern.ucsd.edu"
PHONEBOOK2=Freddy:freddy.fredsmachine.fredsdomain
```

1.44 exptooltypes

```
* * * * *
* * * *
* * * FOR HACKERS ONLY!!! * * *
* * * *
* * * * *
* * * * *
```

If you can't get AmiPhone to sample correctly from your digitizer using any of the "standard" sampler types from the Samplers submenu, there are some extra startup arguments you can use in order to manually control AmiPhone's dealings with your digitizer. The startup arguments below will have effects only when the sampler type is set to CUSTOM in the Samplers submenu.

Note that this will only work with digitizers that connect to the

parallel port (so don't try it with your Toccata board).

Most parallel port digitizers use three binary switches to control the sampling hardware. These switches are called SELECT, BUSY, and PAPER (because they were originally meant to control a printer).

To control the state of these switches, you can set any of several startup arguments. Each of the arguments listed below takes a command string, which is made by concatenating one or more of the following two-character codes:

+s = set the SELECT bit.

-s = clear the SELECT bit.

+p = set the PAPER bit.

-p = clear the PAPER bit.

+b = set the BUSY bit.

-b = clear the BUSY bit.

+i = use the inverting sampler interrupt.

The "+i" argument is only useful when used with the CUSTSTARTBITS command, and causes AmiPhone to use a sampling interrupt function that inverts the waveform it samples (it has the same effect as invoking the INVERTWAVEFORM startup argument)

The ToolTypes that can be used with these command strings are as follows:

CUSTSTARTBITS = Executed when the sampler is enabled.

CUSTSTOPBITS = Executed when the sampler is disabled. (*)

CUSTLEFTBITS = Executed when the left channel is selected.

CUSTRIGHTBITS = Executed when the right channel is selected.

CUSTEXTBITS = Executed when "line" is selected as a source.

CUSTOMICBITS = Executed when "mic" is selected as a source.

CUSTDIRBITS = To set the "data direction" of the bits. (*)

(By default, all three are set to output mode)

(*) = Arguments with an * next to the are probably not useful in getting a digitizer working, but they are included for completeness.

Also, you can use the CUSTSAMPLEADDRESS argument to choose what byte of memory AmiPhone will sample from during each sampling interrupt, when the CUSTOM sampler option is selected. This value defaults to 0xbfe101, which is the address of the parallel port.

If you run AmiPhone from the CLI, debugging information will appear on stdout, informing you of what AmiPhone is doing with

the parallel bits.

So, for example, to exactly mimic AmiPhone's specification for controlling (my version of) the GVP DSS8 sampler, you could run

AmiPhone with these arguments:

```
AmiPhone SAMPLER=CUSTOM CUSTLEFTBITS=+s+p CUSTRIGHTBITS=-s+p
```

or, to emulate AmiPhone's procedure for the Sound Magic, run:

```
AmiPhone SAMPLER=CUSTOM CUSTLEFTBITS=-p+s CUSTRIGHTBITS=-s+p  
CUSTMICBITS=-b CUSTEXTBITS=+b
```

or, to manually set AmiPhone to sample from the Aura's PCMCIA hardware address (left channel):

```
AmiPhone SAMPLER=CUSTOM CUSTSAMPLEADDRESS=0xA20000
```

With these codes, you can try to find out what bit patterns

your sampler works best with. Furthermore, once you've found out the optimum codes, you can send them to me and I will "hard code" them into AmiPhone as another digitizer type.

1.45 maxsampleratearg

With this argument, you can set the upper bound of the sampling rate **slider** to any value between 1,600 and 99,999 samples per second. Be careful, though--actually trying to sample at the high rates that use of this argument can allow may lock up your computer! Only sample at high rates if you have lots of CPU and network bandwidth available.

The default maximum sampling rate is 9,999 samples per second.

Usage Example:

```
AmiPhone MAXSAMPLERATE=12000
```

1.46 maxxmitdelayarg

With this argument, you can set the upper bound of the packet transmission interval **slider** to any value up to 5,000 milliseconds. Note that AmiPhone loses its "real-time" qualities as the sampling delay increases, and that many types of network connection may drop the UDP audio packets if they become too large.

The default maximum packet interval is 700 milliseconds.

Usage Example:

```
AmiPhone MAXXMITDELAY=1000
```

NOTE: At this time MAXXMITDELAY is buggy--if you use it to transmit very long packets, you may experience crashes, lack of transmission, etc.

1.47 techniquearg

Since the Amiga has no dedicated sampling hardware, sampling the parallel port can be rather CPU intensive. One problem in particular that I've experienced is that the sampling interrupt can cause the serial device to be locked out, causing packet reception to cease while the sampler is enabled. To try to circumvent this problem, I've implemented several sampling techniques:

- Hardware Interrupt. This is the default. A CIA timer is allocated, and it causes n interrupts per second, and the sampling is handled within these interrupts.

Usage Example: (not useful, as it's the default, but it's here)

```
AmiPhone SAMPLETECHNIQUE=HARDWAREINT
```

- Software Interrupt. This works essentially like the Hardware Interrupt option, only the CIA interrupt does nothing except generate a software interrupt. The software interrupt then handles the sampling. The advantage of this is that less time is spent in a high priority interrupt, allowing the serial device more CPU. The disadvantage is that there are now twice as many interrupts generated, which means that the maximum sampling rate is halved.

Usage Example:

```
AmiPhone SAMPLETECHNIQUE=SOFTWAREINT
```

1.48 inputsourcearg

Certain digitizers (to be exact, the Sound Magic and Toccata digitizers) have multiple sound input jacks. This startup argument allows you to set the default jack to receive input from.

Usage Example:

```
AmiPhone INPUTSOURCE=MIC
```

or

```
AmiPhone INPUTSOURCE=LINE
```

1.49 amplifyarg

This argument allows you to set the default **Digital Amplification** setting to be used by AmiPhone. Legal settings are 1X, 2X, and 4X. (The 'X' is optional)

Usage Example:

```
AmiPhone AMPLIFY=2
```

or

```
AmiPhone AMPLIFY=4X
```

1.50 holdtotransmitarg

This switch causes the "Hold to Transmit" option in the Transmit Enable submenu of the **Settings Menu** to be selected on startup.

Usage Example:

```
AmiPhone HOLDTOTRANSMIT
```

1.51 sendpriarg

This argument lets you specify the Exec task priority of the AmiPhone client (i.e. of the sound transmitting code). It's a little more convenient than using PriMan (or whatever) to set it. The default priority is the priority of the shell AmiPhone was started from, or 5 if started from Workbench.

Usage Example:

```
AmiPhone SENDPRI=6
```

1.52 receivepriarg

This argument lets you specify the Exec task priority of the local AmiPhoned server associated with this AmiPhone client (i.e. of the sound receiving and playing code). It's a little more convenient than using PriMan (or whatever) to set it. The default priority is 0.

Usage Example:

```
AmiPhone RECEIVEPRI=6
```

1.53 maxbandwidtharg

With this argument you can tell AmiPhone what the maximum bandwidth of your network connection is. AmiPhone uses this information to scale the **bandwidth graph** to the proper height.

The value given in this argument will be the largest value visible in the bandwidth graph. The default value for this is 2880 (the bandwidth of a 28.8kbps modem).

Usage Example:

```
AmiPhone MAXBANDWIDTH=9000
```

(Tells AmiPhone to display a graph with room to display transmissions of up to 9000 bytes per second)

1.54 sampleratearg

This argument lets you choose the default number of samples you will grab from the digitizer per second. The higher this number is, the better the sound you transmit will sound, but the more bandwidth and CPU power it will take. This value can be adjusted at any time with the [Samples per Second slider](#) .

The default sample rate is 5600 samples per second.

Usage Example:

```
AmiPhone SAMPLERATE=6000
```

1.55 xmitonplayarg

This switch, if specified, will cause AmiPhone to transmit to your peer any message or sound file that you are playing back through the [Message Browser Window](#) or [Play Sound File](#) features.

Of course, you must be connected for this to happen.

Usage Example:

```
AmiPhone XMITONPLAY
```

1.56 enableonconnectarg

This switch, if specified, will cause the microphone to become enabled as soon as you connect. It saves you from having to click on the icon after you're connected.

Usage Example:

```
AmiPhone ENABLEONCONNECT
```

1.57 Specify how often to transmit a packet

With this argument you can set the amount of time (in milliseconds) that AmiPhone will sample sound for before it sends out a packet of data. The default is 300 (i.e., send out a packet every 300 milliseconds). You can set this value as low as 90, or as high as 700.

Usage Example:

```
AmiPhone XMITDELAY=200
```


1.58 Specify the minimum volume (threshold)

This argument allows you to set the volume threshold above which AmiPhone will send a sample. This number can range from 0 to 130, with 0 meaning that AmiPhone will always send data, even if the audio stream from the digitizer is silent, and 130 meaning AmiPhone will never send anything no matter how loud it is. The default value is 7.

Usage Example:

```
AmiPhone THRESHVOLUME=25
```

1.59 Specify a compression algorithm to use

This argument allows you to select the **compression algorithm** that will be enabled when AmiPhone starts up. You have a choice of ADPCM2, ADPCM3, or NONE.

Usage Example:

```
AmiPhone COMPRESS=ADPCM2
```

1.60 Specify a computer to connect to on startup

This argument allows you to specify the hostname of another Amiga to connect to on startup. Specifying this keyword has the same effect as starting AmiPhone, selecting "Connect" from the **TCP menu**, and entering the same hostname in the requester.

Usage Example:

```
AmiPhone CONNECT=jfriesne.extern.ucsd.edu
```

If you specify a hostname as the first argument on the command line, the CONNECT keyword is not necessary. Thus,

```
AmiPhone jfriesne.extern.ucsd.edu
```

is equivalent to the first example above.

1.61 Specify a Public Screen

This allows you to specify the name of a Public Screen upon which you want the AmiPhone window to appear. If a Public Screen with the given name does not exist, AmiPhone will fall back to the default public screen.

Remember, with Public Screen names, alphabetical case counts!

Usage Example:

```
AmiPhone PUBSCREEN=MyPubScreen
```

Note: This argument, when specified as a ToolType in AmiPhone's icon, will also be used by AmiPhoned as the preferred screen to open AmiPhoned's info bar on.

1.62 Specify window X-coordinate

This allows you to set the horizontal position of the AmiPhone window.

Usage Example:

AmiPhone LEFT=60

will make the AmiPhone window appear 60 pixels from the left of the screen.

1.63 Specify window Y-coordinate

This allows you to set the vertical position of the AmiPhone window.

Usage Example:

AmiPhone TOP=60

will make the AmiPhone window appear 60 pixels from the top of the screen.

1.64 AmiPhone menus

AmiPhone has a few menus for you to play with...

Project - The usual

TCP - Network connection and disconnection

Messages - Listen to saved messages

Settings - Other random stuff

1.65 messagesmenu

Play Sound File - Opens a File Requester to let you listen to a sound file or a message file that was saved by AmiPhoned.

Note: AmiPhone message files and IFF-8SVX samples will play back at the speed(s) they were recorded at, other files will be played back as raw data at the current sampling rate.

Messages... - Opens the **Message Browser Window** .

Record Memo - This menu item allows you to record a message to yourself, for later reference. Selecting this menu item will cause it to become "checked", and the microphone button will become usable. After selecting this menu item, you may enable the microphone and speak to record a message onto your hard drive. To end the message, select this item again (and the check mark will disappear). This item is usable whether or not you are connected to anyone.

To hear the message you recorded, select "Messages...", above.

1.66 Project Menu

About - Opens a little info-window about **me** and the program.

Quit - Makes AmiPhone go away.

1.67 TCP Menu

This menu controls connecting and disconnecting to a remote Amiga that has AmiPhone installed.

Connect - Connect to a given host. Note that you don't need to specify the user's name at that host, just the computer's name.

When you connect to a remote Amiga, the user at that Amiga will be given the choice of denying your request, receiving your audio stream, or receiving your audio and launching an AmiPhone task of his own to send you audio as well.

Connect To - This submenu allows you to connect to any of ten Amigas without having to type in their IP addresses each time.

All you need to do is select the name that you wish to connect to. Note that this menu cannot be altered while AmiPhone is running--you must use the **PHONEBOOKx** ToolType or Command Line Argument to specify name/IP address pairs before starting AmiPhone.

Disconnect - Sort of the Internet equivalent of hanging up, I suppose.

Show Daemon - If a AmiPhoned daemon corresponding to this client is present, this option can be used to make it open or close its GUI interface. If the Daemon interface is currently open, this option will be "checked".

This menu is only enabled if you have AmiTCP running when AmiPhone is started.

1.68 messagebrowserwindow

The message browser window allows you to listen to any saved voice messages that you might have. It can be called via the "Messages..." item in the **Messages** menu. It is only available if you have **defined a Voice Mail directory** .

When opened, the Message Browser Window presents you with a ListView of saved messages. Each item in the ListView contains information such as the machine name that sent the message, the length of the

message in seconds, and the time and date that the message was received.

To hear a message, you either double-click on it, or click on it once and then click the "Play" button. To stop a playing message, you may click the "Stop" button.

The "kill" button will erase the message from the disk. Be careful, as the message file will be deleted immediately, and once it is gone, it's gone for good.

The "scan" button tells AmiPhone to re-scan the Voice Mail directory and update its displayed message list to match what is there.

To exit the Message Browser, click "exit", or the close box of the window.

1.69 Settings Menu

There are currently three items in this menu:

Sampler - A submenu that allows you to specify which audio sampler you are using. If your sampler isn't listed, then your best bet is Generic.

Compression - A submenu that allows you to choose which **compression algorithm** you wish to transmit data with.

Transmit Enable - A submenu that allows you to choose the behavior of the **microphone button**. If "Toggle" is selected (default), clicking on the microphone button will cause it to switch between active and passive states. If "Hold to Transmit" is selected, the button will only remain active until you release the mouse button.

Input Gain - This submenu allows you to raise and lower the gain of the sound signal entering the digitizer. This menu currently only works with the DSS8, Toccata and PerfectSound digitizers.

Digital Amplify - This submenu lets you **digitally amplify** the input signal that AmiPhone is digitizing. The Toccata board does not support this feature.

Input Channel - This submenu lets you choose which input channel to sample from--left or right. This menu may not work with all digitizer types. You can set the default for this menu with the **INPUTCHANNEL** ToolType.

Input Source - This submenu lets you choose which input jack to sample from--either the microphone jack or the line-level jack.

This submenu is only applicable to the Toccata and Sound Magic digitizers.

Enable on Connect - If this is checked, you will begin transmission of sound over the network as soon as you are connected to your partner. If it isn't checked, you will have to click on the microphone button to enable it before you can start talking to your buddy. The default state of this menu item can be set via the **ENABLEONCONNECT** startup option. If you are using "Hold to Transmit" mode, this option is not applicable.

Xmit on Play - If this is checked, any saved AmiPhone messages or sound files that you play using the **Message Browser Window** or **Play Sound File** options while connected will be sent to your peer's speakers as well. The default state of this item can be set via the **XMITONPLAY** startup argument.

TCP Batch Xmit - If this is checked, AmiPhone sound data will be sent via TCP rather than UDP, and will not be played on the receiving end until the entire message is received. This will cause extra delay before the other person hears your voice, but on too-slow lines, it will go a long way towards making the sound error-free when it is played.

1.70 Sound Transmission technique and limitations

Note that you will need to use a compression algorithm that fits within the bandwidth of your connection, if you want a steady stream of audio. If you and your partner plan to be talking at the same time (full duplex) you will need an amount of bandwidth that is greater than or equal to the sum of the two compression algorithms you are using (yes, you can each be using different algorithms, and even different sampling rates).

None - Transmits the sound with no compression. Don't use this unless you are connected to localhost or over some other very fast link! (Sounds very good though)

ADPCM2 - Gives 4:1 compression. With this, for example, you should be able to transmit sound 1-way (at a time) over a 14.4 modem at 5600 samples per second.

ADPCM3 - Gives 8:3 compression. Sounds a bit better than ADPCM2, though.

1.71 phoneutil

PhoneUtil is a little CLI-only program that is included with AmiPhone to allow the user to do file conversion between AmiPhone's voice message format and other sound formats.

Right now PhoneUtil can only convert an AmiPhone voice mail message into raw sound data. In the future it will also be able to convert AmiPhone to and from IFF-8SVX format.

PhoneUtil should be given the name of a source file (usually something like AmiPhoneMessage.822965755) and optionally, the name of a destination file (if none is given, AmiPhone will create a name by adding .raw onto the source file name).

1.72 digitalamplification

Sometimes when you are using a microphone as input to your digitizer, the signal coming from the microphone is a little bit soft. You have to talk loudly into the microphone to be heard clearly. For some digitizers, such as the DSS8, PerfectSound, or Toccata, you can use the Microphone Gain **startup argument** or **menu item** to fix this problem. However, other brands of sampler either do not support this, or do so in a way that is not supported by AmiPhone.

The best solution to this problem is to go and buy an op-amp, and use that to amplify the signal in an analogue fashion. However, that requires money, so you might not want to do that. :)

The next-best solution is to amplify the signal after it has been digitized, as this can be done in software. The signal can be amplified by a factor of two just by shifting the bits of each sample left one position. AmiPhone has the ability to do this for you. Just select "2X" or "4X" from the Digital Amplify submenu, and each sample collected will be shifted left 1 or two positions, respectively.

The downside to this is that shifting the bits left significantly lowers the largest sound that can be sampled, causing noise if you speak too loudly. This is because the most significant bits are lost during the bit-shift.

1.73 situation

When emailing a bug report, please let me know what kind of Amiga you have, what operating system version, which type of digitizer, and exactly what must be done to reproduce the bug (e.g. what actions did you take to make it appear?) If the bug involves sampled sound quality, feel free to record snippets of the sound to files using the "Record Memo" function of AmiPhone and uuencode the resulting files to me.

1.74 Thanks

Thanks go to the following people:

All the beta testers who helped me debug this program.

Martin Brenner for Audio-Handler, which helped considerably with the debugging and early versions of the code.

Alex Smith for the source to AGMSRecordSound, which served as a good example for the PerfectSound recording routine.

Matt Dillon for DICE.

Frank Wille for PhxAss.

Commodore for the nifty computer and O/S.

NSDi for AmiTCP.

My friends, and all the nice people on the Internet who helped me out with Amiga problems.

1.75 faq

Q: Where can I find someone to talk to with AmiPhone?

A: There is often a #amiphone channel active on EFNet IRC, that's your best bet. Also try other IRC channels like #amiga.

Q: My **input volume indicator** registers high volume, even when there is no input signal. What can I do about this?

A: If you are using a parallel port sampler, try the **INVERTWAVEFORM** startup argument. This may fix the problem. If you are using an AHI or Toccata digitizer, then this is for now the correct behavior. I may in the future implement volume detection for these boards, but I want to get the basic functionality working before I start on that.

Q: How can I get to the AmiPhoned GUI?

A: If the AmiPhoned you wish to access is paired with an AmiPhone client, you may select "Show Daemon" from that AmiPhone client's TCP menu. Other than that, the best way is to use the "Break" AmigaDos command or your favorite system utility to send a CTRL-E to the AmiPhoned process. Also see the **SHOWDAEMON ToolType** .

Q: Is AmiPhone compatible with (IPhone/VoiceChat/NetFone/Whatever) ?

A: No, for now AmiPhone is only compatible with AmiPhone. It may at some time in the future be compatible with something, but it will probably not ever be compatible with IPhone (IPhone apparently uses IRC, which is not at all how AmiPhone operates).

Q: Whenever I start sampling with AmiPhone, my machine locks up.

A: You probably are sampling at a faster rate than your CPU can handle. Try sampling first at 1600 bytes per second, and slowly increasing the sampling rate until you find your maximum comfortable speed.

Once you find this maximum speed, you may want to set the **MAXSAMPLERATE** ToolType to this, to prevent accidental lockups in the future.

Q: What samplers does AmiPhone work with?

A: AmiPhone should work with any standard 8-bit sampler that attaches to the parallel port. AmiPhone was written using a GVP DSS8, and has been tested successfully with the DSS8+, PerfectSound 3, Sound Magic, and several homemade samplers. AmiPhone also supports the Toccata Zorro II sound board and the Aura PCMCIA sound card for digitizing purposes, but these have not been well tested. If you have a Toccata or Aura, please tell me how well (if?) it works!

Q: But I don't have any sampler!

A1: Well, go out and buy one, then. ;^)

A2: There are some schematics and instructions on Aminet on how to make your own "homebrew" sampler, that can be used with AmiPhone's **GENERIC** sampler setting. They are listed under /hard/hack/audiodig.lha.

Q: I can't get AmiPhone to work with my sampler!

A: There are probably problems with AmiPhone and some samplers; that's because I only have a DSS8 here to use myself, and I must rely on other people to test the code for other brands. If you can't get your sampler working, the thing to do is this: Copy the included `amiphoned_debug` daemon to `amitcp:serv`, and then run AmiPhone from the CLI. Select your sampler from the "Sampler" menu, and connect to localhost. A window should pop up on your screen with debugging output from AmiPhoned. Try sending sound to yourself, and copy and paste any output from the `AmiPhoned_debug` window and the CLI you

started AmiPhone from into an email to **me** . Also describe exactly what is going wrong (i.e. no sound output, wrong channel, etc.) This will help me in fixing the bugs. If you wish to experiment further, try using the CUSTOM sampler type and the **CUSTOM startup arguments** to manually control AmiPhone's interface to the parallel port.

Q: Is AmiPhone's source code available?

A: Yes. I will email the source code to you if you send me a **donation** and/or ask nicely. :) If you use it for a project, I ask that you give me a credit line in your docs.

Q: What does 'bp' stand for on the Threshold Volume Slider label?

A: I don't know. It just looks good. :)

Q: How can I make my voice easier for the other guy/girl to understand?

A: There are several issues that can affect sound quality. The first, and simplest, is that your digitizer setup may be wrong. Connect to localhost and try your digitizer at various settings to see what you are sounding like. The second issue is sample rate; the lower the sample rate is, the worse you sound (high frequencies in your voice are lost). However, if you raise the sample rate too high, then the information transmitted may become greater than the bandwidth of your connection, in which case the listener will hear gaps of silence at intervals when you are speaking. This makes it hard to understand you. However, there IS a way to have your cake and eat it too; if you don't mind a little additional delay between when you speak and when you are heard, you can select **Batch TCP Xmit** from the Settings Menu, and AmiPhone will buffer your voice and not play it back to the other person until you are done speaking. ("done speaking" means when the digitizer goes quiet or is disabled, so for the Toccata and AHI boards you should disable the microphone button when you are done speaking, because they cannot detect when you are quiet.)

1.76 The ground was littered with squashed bugs...

("-" = new feature, "*" = bug fix)

1.80 : (Public Release 5/12/96)

- Started to add AHI sampling support. (NOTE: AHI support isn't working yet, so it is not selectable)

- Added a Record toggle to AmiPhone's **Project Menu**

- Added transmission-side queueing of TCP data. This virtually guarantees that in Batch TCP Mode, data will not be dropped due to a slow connection. (Unless your Amiga runs out of memory)
- Changed the default "idle" sampling rate from 300 to 200 Hz.
- EditText.rexx now replaces the original file if the file modification failed.
- * Finally found and killed the bug that was causing the AmiPhone sampling routine to interfere with serial port operation. It should now be possible to leave the sampler enabled all the time, with no adverse effects.
- * The ASL FileRequester now uses its own IDCMP port, which fixes problems some people were having with interference between it and the main process's IDCMP.
- * The session key scheme was only working for the connecting party. Now it should work for the receiving party as well.
- * The Toccata board support is in the process of being improved, so it may work better now. Then again, it may not.

1.71 : (Public Release 5/5/96)

- Added the **IDLERATE** startup argument.
- * The session key transmission *still* didn't work right. Fixed.
- * Fixed the "Hold To Transmit" mode, which broke under v1.70.
- * Made the About requesters a bit prettier.
- * Made the Graphics Daemon code a bit less thread-unsafe.
- * Fixed a bug that sometimes caused the sampler not to return to its idle frequency when sound stopped.

1.70 : (Public Release 5/3/96)

- AmiPhone now samples the parallel port at only 300Hz by default, and automagically bumps the sampling frequency up when sound is detected. This saves lots of CPU time when you aren't saying anything.
 - AmiPhone now compares version numbers during connections, and if the version numbers don't match in the .1's place column, displays a warning requester to the party who needs to upgrade.
 - Enhanced the Installer script and EditTextFile.rexx so that if the config file editing fails, the user is notified.
 - * Wiped out a completely stupid bug that was causing AmiPhone's session key transmission scheme to break when the remote host was not "localhost". Connections should be more straightforward now, and the "Show Daemon" option and etc. should work.
-

* Fixed a bug that caused AmiPhone not to parse CLI arguments if it couldn't find an AmiPhone icon.

1.6 : (Public Release 4/27/96)

- Added the **TCP Batch Xmit** menu item.
- Added the **TCPBATCHXMIT** startup argument.
- The Browser window and "Play Sound File" menu item can both play (and transmit) IFF-8SVX and raw sound data files now.
- If there is no FileNote for a file in the Voice Mail directory, the Browser window will display the file's name.
- * Rewrote the IPC code to be simpler and hopefully more robust.
- * The AmiPhone window now waits for you to close the File Requester window before exiting.
- * "Play Sound File"'s file requester now remembers the directory you last played a sound from and returns there when you open it the next time.
- * When you disable the sampler, AmiPhone now transmits the valid portion of the buffer that was being sampled into at the time.

1.5B : (Public Beta Release 4/12/96)

- Added support for the Aura PCMCIA sampler.
- Added the **CUSTSAMPLEADDRESS** startup argument.
- Added the **MICGAIN** startup argument.
- Added input gain control for the DSS8 digitizer.
- * Handling of \$HOSTNAME should be more reliable. AmiPhone now calls gethostname() instead of using getenv("HOSTNAME").
- * ADPCM JoinCodes are now explicitly transmitted with each data packet, rather than being recomputed on the receiving end. Before, if a data packet was lost, there was no way to recover the succeeding packet's JoinCode, leading to poor sound quality after a dropped UDP packet.

1.41B: (Public Beta Release 3/12/96)

- * Oops! I left some debug output turned on in 1.4B, which caused AmiPhone to be slow to the point of unusability. Mea culpa!
- * Kludged around a bug that caused Software Failures when running with some KickStart 3.x setups

1.4B : (Public Beta Release 3/8/96)

- Added the **Custom** sampler type, allowing users to **hack** AmiPhone's sampler interface a bit if they so choose.
 - Added the **INVERTWAVEFORM** startup argument.
 - Optimized the sampling interrupts a bit more, and cleaned up
-

the assembly code. (Thanks to Johan Torin for help with this!)

- Added the **PhoneUtil** utility to the distribution.
- Added "Connect To" submenu to the **TCP menu** .
- Added **PHONEBOOKx** Startup Arguments.
- Added the **MAXXMITDELAY** Startup Argument.
- * Fixed a bug which caused AmiPhone to "Software Failure" if you recorded a memo while AmiTCP wasn't running

1.3B : (Public Beta Release 1/27/96)

- Went and redid the connection code AGAIN! Now AmiPhone uses tcp streams to connect and to send control data, while still using udp packets to send the actual sound.
- * Playing a voicemail message no longer disables sampling unless you have "XMit on Play" selected.
- * Hopefully fixed a bug where some digitizer settings were not retained if you enabled, disabled, and re-enabled sampling.

1.2B : (Public Beta Release 1/5/96)

- AmiPhoned now only allocates an audio channel when it actually has sound data to play; and it deallocates it whenever it doesn't. This means that you can now run more than 4 AmiPhoned's at once! (Although not more than 4 will be able to play sound at any given moment!)
- Added an error signal ('X') to the AmiPhoned title bar.
- * Fixed a stupid bug that caused the AmiPhoned title bar to be empty if it wasn't set to open immediately.
- * Relay daemons will now open a title bar by default. (unless SHOWDAEMON is explicitly set to NO)

1.1B : (Public Beta Release 1/5/96)

- Added the "Show Daemon" toggle to the **TCP Menu** .
- Added **Relays** to AmiPhoned!
- Added **menus** to the AmiPhoned title bar.
- AmiPhoned title bar now is sized to fit the text it displays.
- Added a flashing asterisk indicator to the AmiPhoned title bar.
- * Fixed a bug that caused AmiPhoned to not take a message when you clicked on "Take a Message". Sigh....
- * Rewrote the AmiPhone<->AmiPhoned IPC code to be more reliable.
- * Month was being expressed in the FileNote as [0..11]. Now [1..12].

1.0B : (Private Beta Release 1/3/96)

- Now checks to make sure a Toccata board exists instead of just assuming that having toccata.library implies a board is available.

-
- Added the **Digital Amplify** submenu to the Settings Menu.
 - Added the **AMPLIFY** and **INPUTSOURCE** startup arguments.
 - Added "Show Daemon" option to the **TCP menu** .
 - * Fixed some minor stupidities in the docs.
 - * Rewrote the interrupt handler to take less cycles
 - * The Perfect Sound interrupt handler should now record all 8 bits of each sample.
- 0.9B : (Public Beta Release 12/16/95)
- Added the **Xmit Delay** and **Thresh Vol** sliders.
 - Made the main window wider, and changed the layout a bit.
 - Added support for the "Sound Magic" digitizer
 - Added the **Input Source** submenu to the settings menu.
 - Added the **Record Memo** option.
 - Added the **Xmit on Play** option, and the **XMITONPLAY** startup argument.
 - A prettier icon, courtesy of Jon-Eric Eliker and Martin Huttenloher.
 - Added the "Scan" button to the **Message Browser Window** .
 - * Rewrote the IPC code for the sound player and browser window. Now they use messages instead of just signals. Should be somewhat more reliable now.
 - * Renamed the ToolType "PACKETINTERVAL" to **XMITDELAY** .
 - * Renamed the ToolType "MINVOLUME" to **THRESHVOLUME** .
 - * Renamed the ToolType (and menu item) "XMITONCONNECT" to **ENABLEONCONNECT** .
 - * Maximum volume threshold is now 130 instead of 255.
 - * Fixed a bug that could cause unterminated strings to be created.
 - * Fixed a couple of memory leaks.
 - * The Message Browser window's minimum size should now always be large enough to draw the ListView and button gadgets.
 - * Hacked around the bug that was causing garbage to be appended to voicemail filenote lines.
- 0.8B : (Public Beta Release 11/22/95)
- Added the **Message Browser Window** .
 - Completed basic voice mail features. Added **VOICEMAILDIR** , **MAXVOICEMAILSIZE** , **MAXMESSAGE SIZE** , and **AWAYVAR** Tool Types.
 - AmiPhone can now be started without AmiTCP running. The TCP Menu will be disabled, however.
 - Installer script now can setup voice mail for you.
 - * **Play Sound File** menu item now works for hearing saved messages.
- 0.7B : (Beta Release 11/12/95)
-

- AmiPhoned can now optionally open a little title bar, so that you can disconnect it by clicking on its close box.
- Added **DAEMONLEFT** , **DAEMONTOP** , and **SHOWDAEMON** ToolTypes.
- AmiPhoned now obeys the **PUBSCREEN** ToolType when applicable.
- Absolute maximum sampling rate is now 32767 bytes per second, and the problem with the sampling rate readout overwriting the performance graph is fixed.
- Beginnings of Toccata board support. Does it work? Who knows?
- Beginnings of voice mail!
- Added **Messages menu** .

0.6B : (Beta Release 11/05/95)

- Interrupt handler now faster: uses pointer math instead of array math
- Added **SAMPLER** Startup Argument. Hopefully this will allow AmiPhone to work with PerfectSound, etc.
- Added Left/Right input channel **menu options** , and their corresponding **ToolTypes** .
- Added Raise/Lower input gain **menu options** .
- Slightly more verbose debugging messages in AmiPhoned.
- Installer script now prompts for what type of sampler you have, and sets the SAMPLER ToolType accordingly.
- Installer now detects if HOSTNAME isn't set and offers to let you set it.
- AmiPhone now reads ToolTypes from its icon even when started from the CLI. CLI arguments still override ToolTypes, of course. The only ToolType that is ignored from the CLI is CONNECT, to avoid nasty surprises...

- Added **Sampler** submenu to the Settings menu.

* Now handles connects to "localhost" by translating it to \$HOSTNAME if \$HOSTNAME is set.

0.5B : (Public Beta Release 10/15/95)

- Added **MAXSAMPLERATE** Startup Argument.
- Added ability to tell AmiPhoned where AmiPhone is located via **ENV var** .
- Added one more frame to the "xmitting" animation.
- Enabling the sampler on an AmiPhone session while it is already enabled on another AmiPhone session will now cause the other AmiPhone session to disable its sampler so that we can use it. (Interprocess communication is so much fun!)

- Added a Zoom gadget.

* Fixed a graphic glitch that could occur if you disabled the microphone

while talking, then enabled it again.

- * Fixed the microphone volume glitch that occurred when changing sampling rates.
- * The "Xmit on Connect" menu option is now disabled when you are using "hold to transmit" mode. Also fixed it so that it doesn't send a sound packet when connecting in this state.
- * Bandwidth meter is now updated only when AmiPhone is actually connected.
- * Connect string requester is now half the screen size, rather than two thirds of it.

0.4B : (Beta Release 10/12/95)

- Added **SENDPRI** and **RECEIVEPRI** options.
- Added **SAMPLETECHNIQUE** startup option.
- Added **Transmit Enable** submenu and **HOLDTOTRANSMIT** startup option.
- Changed the default **MAXBANDWIDTH** to 2880. (1440 was claustrophobic)
- * Finally found and squished the "Enforcer hit per packet" bug.
- * Hopefully fixed the bug that would cause the occasional random line to be drawn across the window.
- * AmiPhoned's task name no longer has a carriage return in it.
- * Recoded the CIA timer code from Commodore's RKM example.

0.3B : (Beta Release 09/24/95)

- Added in color-code logging of errors on the **bandwidth graph** .
 - The parallel port is now allocated only when needed, rather than at startup. It is deallocated whenever you are not sampling.
 - You can now run multiple AmiPhone sessions at once--up to four AmiPhoned's can run at once (one per sound channel), and any number of AmiPhone clients may run at once, although only one may be sampling at any given time.
 - The title bar now shows who (if anyone) you are connected to.
 - Added some **keyboard shortcuts** .
 - Added **MAXBANDWIDTH** startup option.
 - Now uses internal messaging for key transmission instead of an environmental variable.
 - * Fixed the bug that wouldn't let AmiPhoned be connected to more than once per AmiTCP session.
 - * Fixed an embarrassing typo--the connection requester now identifies the program as AmiPhone, rather than AmiSlate. Cut and paste can be SO treacherous sometimes... :)
 - * AmiPhoned should no longer try to set up a connection whenever any
-

packet arrives on its port; instead, it will only respond to "connect" type packets.

* AmiPhoned now responds to a control-C at any time to exit.

0.2B : (Beta Release 09/15/95)

- Now uses udp packets instead of a tcp stream. This should be faster.
- Now uses built in sound routines instead of Audio-Handler.
- Has two forms of audio compression, ADPCM2 and ADPCM3. These should make it definitely usable over a 28.8 connection, anyway.
- Graphics updating is now handled by an asynchronous subtask. This means that choosing from menus, etc. should no longer cause audio transmission to cease.
- Added a last-n samples averaging algorithm to calculate the scrolling **graph** , and added a "bytes sent" color that sits on top of the "bytes received" color.
- Added a microphone volume indicator.
- Added a **samples-per-second slider** .
- Added XMITONCONNECT startup option.
- Added PACKETINTERVAL startup option.
- * Now can be started from a Workbench icon.
- * Removed the 1,2,4,8 bit buttons, as the compression obsoletes them.
- * The docs are more complete and accurate now.
- * The Install script now asks you where you wish the client to be installed, and does things in a bit more straightforward manner than before.

0.1B : (Beta Release 09/06/95)

- Can only be started from the CLI. Lots o' bugs.

1.77 How can I check that everything is okay?

If you want to make sure that you are transmitting okay, or just see what different transmission options sound like, start AmiPhone, and connect to "localhost". This will allow you to transmit sound to yourself.

1.78 What's Next?

Note: These are things I'm thinking of implementing; Whether I actually implement them or not depends on how difficult they will be to implement and user response (both in the form of **communications** and **donations**).

- Make AmiPhone compatible with some of the similar programs available for other platforms (SpeakFreely, NetFone, VoiceChat, PGPPhone, etc.)
- Augment PhoneUtil.
- More and better compression algorithms!
- An ARexx port. Would this be useful?
- Allow an outgoing message to be recorded so that callers hear it before leaving voice mail (instead of just seeing a requester).
- Use file notification to update the message browser screen whenever a new file is received.
- Iconify. The icon could change image to indicate pending messages...
- Make the GUI font-sensitive.

1.79 Known Bugs and Other Problems

Here are the things that still don't work right with AmiPhone V1.80:

- There are occasional skipped or repeated packets.
- Using the MaxXmitDelay arg to set the packet sizes to large values (e.g. greater than 1500 milliseconds) causes big problems (crashes, etc.)
- If AmiPhoned exits while there are still audio packets ready to be read from the stream, their memory is not freed.
- The Toccat support is very iffy, and the AHI interface is not implemented yet.

1.80 otherprogs

Other Amiga programs I have written (both require AmigaDos2.04 or higher):

GadMget - Loads in an Aminet RECENT or INDEX file and lets you choose files to download via a pair of ListViews. Features keyword searching and sorting by name, size, age, directory, and description. When you're done, it outputs the ftp commands that are needed to download the selected files. The output formatting is extremely flexible, allowing generation of many formats: ftp, ncftp, ftp-by-mail, shell scripts, etc.

Comes with an ARexx script to completely automate downloading with ncFTP. (util/misc/GadMget2.05.lha,93K)

AmiSlate - A paint program that works with AmiTCP to allow two people to cooperatively paint on the same drawing from different computers. Features an extensive ARexx port which allows the construction of new features and games. Comes with ARexx scripts for chess, tic-tac-toe, backgammon, and others. (comm/tcp/AmiSlate1.4.lha,115K)
