SOUND EXPERIMENT I ©

DIGITIZE ANY SOUND INTO MEMORY INCLUDING
SPEECH, AND PLAYBACK IN AMAZING FIDELITY

DATA FORCE INCORPORATED
HINSDALE, ILLINOIS

REQUIRE: MINI MEMORY

CONSOLE BASIC SUPPORT FOR ALL
SOUND EXPERIMENT I FUNCTIONS
ALLOWS USER TO EXPERIMENT FUL-
LY WITH ONLY CASSETTE & MINMEM

DATA FORCE INCORPORATED
10 S. 312 Hampshire Lane East
Hindsdale, IL 60521
(312) 323-0179

SOUND DIGITIZER EXPERIMENT 1
REQUIRES MINIMEMORY
CASSETTE RECORDER

<u>Commands</u>
CALL LINK("ON1") or "ON2"

Turn on cassette 1 or cassette 2 respectively,that is activate remote control to <u>on</u> state.

"OFF1" or "OFF2"

Turn remote control to <u>off</u> state on cassette 1 or 2 respectively

"OFFSND" "ONSND"

Do (or do not) route sound recieved by computer from cassette input to the computer controlled amplifier (TV or Monitor)

"RECORD"

Digitize sound picked up from jack plug to cassette <u>speaker</u> jack.Begin after first sound and untill .5 second pause.

"PLAY"

Recreate sound signal from digitized data and put output to sound generator of the computer.

"SNOOP"

Listen for sound and return immediately after first sound is heard.

"GET"

Return digitized data to basic array designated.

"SAY"

Return array to digitized area and execute "PLAY".

"GETPAT"

Return selected pattern from digitized area to basic string.This pattern should be representative of the first sound in digitized area.

<u>SOUND EXPERIMENT 1</u>
The objectives of the Sound Experiment 1 program are three fold

1) To give the user access to the cassette ports and other facilities not accessable from Console Basic or Extended Basic.

2) To provide the routines to digitize any sound, using the cassette player as a pre-amplifier , and the sound generator for playback.

3) To provide a number of Console Routines illustrating the use of the Sound Experiment 1 functions.

PRELIMINARY CONSIDERATIONS

Since we will use the cassette mircophone and pre-amp with the program, the user must determine what he must do to have the open microphone we will need. The microphone and the pre-amp will be active on all recorders PLAY and RECORD (record mode) are on. Since we will not record on tape, user should determine if his microphone stays open when pause is used. If it does, PLAY/RECORD/PAUSE would be a preferable setting for most of our examples. If you are fortunate enough to have a loudspeaker setting (direct mike to speaker) you sould be able to use that setting.

Our asumption will be that your PLAY/RECORD/PAUSE setting will work, if it does not load a scrap tape and run without pause on.

Since we will be recording from audio input, and not computer program output, run with the red wire (MIKE JACK) disconnected,this will make the internal microphone active. If there is no internal mick on your recorder,plug in an external microphone.

Experiment with tone and volume settings for a change in quality of recording, but ALL settings which work for normal audio recording should produce digitized recordings.

STARTING UP

1) Turn off computer.
2) Insert Minimemory module into slot.
3) Turn on computer.
4) Put sound experiment tape into recorder.
5) Any key to obtain menu.
6) Select menu 2 (debug)
7) Enter any key for DEBUG COMMAND MODE.
8) Enter l for LOAD MINIMEM cassette routine.

Your minimemory sould be loaded with the program code by now.*NOTE* There sould be a way of loading the program into the supercart or gramcracker.

SAMPLE PROGRAMS

enter RUN 2110 for program 1
****

```
2110 CALL SOUND(10,44733,1)
!BEST RESULTS IN PLAY!
2120 CALL LINK("PLAY")
!tape
```

```
2220 CALL LINK("OFF1")
2230 CALL LINK("OFF2")
2240 CALL LINK("OFFSND")
!WAIT FOR SOUND/RECORD LOOP
2250 CALL LINK("SNOOP")
2260 CALL LINK("ON1")
2270 CALL LINK("RECORD")
2280 CALL LINK("OFF1")
2290 GOTO 2250
```
****

Sample program #1 SNOOP (turn off 32k or remove if you have it)

SNOOP listens for any sound , then returns to basic, statement #2120 will play any message we left in DIGITIZE MEMORY

****

enter RUN 2450 program #2

****
```
2450 CALL LINK("OFF1")
2460 FREQ=44733
2470 CALL SOUND (10,FREQ,2)
2540 CALL CLEAR
2550 PRINT "SPEAK IN .5 SECONDS"
2560 CALL LINK("RECORD") !digitizes after first sound
2570 FOR Y=1 TO 10
2580 FOR X=1 TO 20
2590 NEXT X
2600 CALL LINK("PLAY")
2610 FREQ=FREQ-4000
2620 CALL SOUND (10,FREQ,2)
2630 NEXT Y !plays back with variance in frequency to show limited
control
```
****

This routine digitizes sound and plays it back through sound generator. You have a limited control of sound quality by setting freq. of first sound VOICE as shown. Best results will be obtained at frequencies above the audible range.

We are not actually GENERATING sound through the sound generator in a normal sense. We are utilizing the fact that the generator will "flutter" the speaker as we alternate or flip/flop the signal we are sending.

The delay or TIME between FLIPS creats our frequency reproduction.

To make some comparisons between this program and other sound digitizing applications, we can <u>SAMPLE</u> the signal based on frequency only at the rate of approximately 3000 times a second.

A HI-FI system useing digitized sound would sample up to 50,000 times a second, also code additional data.

Texas Instruments Professional Computer with its voice recongnition lab samples signals at aproximatly 8000 cycles per second.

We have no way to test volume, or other attributes of the sound, we sample only the very simplest concept of frequency, and reproduce that.Very high sounds (bell or ting of crystal) will sound very flat, and lower than atual. we probaly miss many of the CYCLE changes of the frequencies produced by these sounds.

However, you should experiance GOOD voice reprodution , and be able to distinguish between different voices easily.

```
     ****************************
     enter RUN 2710 (program 3 storing patterns locally in basic arrays
     ****************************
     2710 OPTION BASE 0
     2720 DIM A$(20)
     2730 DIM B$(20)
     2730 CALL SOUND (10,44733,1)
     2750 CALL LINK ("OFF1")
     2760 PRINT "SPREAK IN .5 SECONDS"
     2770 CALL LINK ("RECORD")
     2780 FOR X= 1 TO 100
     2790 NEXT X
     2800 CALL SOUND (10,44733,1)
     2810 CALL LINK ("PLAY")
     2820 INPUT "SAVE WORDS?";ANS$
     2830 IF ANS$="N" THEN 2760
     2840 CALL LINK ("GET",A$())
     2850 PRINT "SAY SOMETHING ELSE"
     2860 CALL LINK ("RECORD")
     2870 CALL SOUND (10,44733,2)
     2880 CALL LINK ("GET",B$())
     2890 PRINT "SAY A$"
     2900 FOR X=1 TO 10
     2910 NEXT X
     2920 CALL LINK("SAY",A$())
     2930 PRINT "SAY B$"
     2940 FOR X=1 TO 10
     2950 NEXT X
     2960 CALL LINK("SAY",B$())
     2970 GO TO 2890
     ******************************
```

While the "RECORD" and "PLAY" routines will use the 32k expansion if it is on,the "GET" and "SAY" routines will not work.

"GET" requrires an array be passed because basic can only pass 255 byte strings,and our digitized sound will easily exceed that. Each sound to be saved must be passed in a seperate array, you can take the data in the array and store it on disk if you wish,or write it to CS2 for later retrieval by useing the SEG$ BASIC command to reformat the data to a record size.

To avoid standing at the copy machine I have typed in the docs for the Data Force Sound Digitizer experiment. I have omitted the listings on the document that show the Basic programs. You can list the program called "BASIC" and refer to it when necessary. This all came on cassette originally and I have changed most of the text accordingly.

Laboriously transcribed by DOUGLAS A. OTTEN

COMMANDS:

CALL LINK("ON1") or "ON2"
TURN ON CASSETTE 1 OR CASSETTE 2 RESPECTIVELY, THAT IS ACTIVATE REMOTE CONTROL TO <ON> STATE.

CALL LINK("OFF1") or "OFF2"
TURN REMOTE CONTROL TO <OFF> STATE ON CASSETTE 1 OR 2 RESPECTIVELY.

CALL LINK("OFFSND") or "ONSND"
DO (or do not) ROUTE SOUND RECIEVED BY COMPUTER FROM CASSETTE INPUT TO THE COMPUTER CONTROLLED AMPLIFIER (TV or Monitor)

CALL LINK("RECORD")
DIGITIZE SOUND PICKED UP FROM JACK PLUG TO CASSETTE <SPEAKER> JACK. BEGIN AFTER FIRST SOUND AND UNTIL .5 SECOND PAUSE.

CALL LINK("PLAY")
RECREATE SOUND SIGNAL FROM DIGITIZED DATA AND PUT OUTPUT TO SOUND GENERATOR OF THE COMPUTER.

CALL LINK("SNOOP")
LISTEN FOR SOUND AND RETURN IMMMEDIATELY AFTER FIRST SOUND IS HEARD.

CALL LINK("GET")
RETURN DIGITIZED DATA TO BASIC ARRAY DESIGNATED.

CALL LINK("SAY")
RETURN ARRAY TO DIGITIZED AREA AND EXECUTE "PLAY".

CALL LINK("GETPAT")
RETURN SELECTED PATTERN FROM DIGITIZED AREA TO BASIC STRING. THIS PATTERN SHOULD BE REPRESENTATIVE OF THE FIRST SOUND IN DIGITIZED AREA.

The objectives of SOUND EXPERIMENT I program are three-fold:

1) To give the user access to the cassette ports and other facilities not accessible from Console Basic or Extended Basic.

2) To provide the routines to digitize any sound, using the cassette player as a pre-amplifier, and the sound generator for playback.

3) To provide a number of CONSOLE ROUTINES illustrating the use of the SOUND EXPERIMENT I functions.

PRELIMINARY CONSIDERATIONS

Since we will use the cassette microphone and pre-amp with the program, the user must determine what he must do to have the OPEN MICROPHONE we will need. The microphone and pre-amp will be active on all recorders when PLAY/RECORD (record mode) is on. Since we need not record on tape, user should determine if his microphone stays open when PAUSE is used. If it does, PLAY/RECORD/PAUSE would be a preferable setting for most of our examples. If you are fortunate enough to have a LOUDSPEAKER setting (DIRECT MIKE TO SPEAKER) you should be able to use that setting.

Our assumption will be that your PLAY/RECORD/PAUSE setting will work, if it does not, load a scratch tape and run without pause on.

Since we are recording from audio input, and not computer program output, run with the MIKE jack (red wire) disconnected, this will make internal microphones active. If there is no internal mike on your recorder, plug in your remote microphone.

Experiment with tone and volume settings for change in quality of recording, but ALL settings which work for normal audio recording should produce digitized recordings.

STARTING UP

1) Turn off computer.
2) Insert MINI-MEMORY module in cartridge slot.
3) Turn on computer.
4) Initiate MINI-MEM with "CALL INIT" command.
5) load object file by entering CALL LOAD("DSK1.DATAFORCE")

Your Mini Memory should be loaded up with program code now.

If program does not load, retry, if this continues to fail return to your dealer or Data Force Incorporated with all original documentation for immediate replacement. (HA! HA! HA!)

There are 4 routines written in BASIC illustrating how to use the function commands of SOUND DIGITIZER EXPERIMENT I. These are in one program on the Diskette and may be used after loading the Mini-Mem.

Each routine can be executed by entering the statement number after the RUN command. The statement number can be found in this documentation with the description of each routine.

SAMPLE PROGRAM #1 "SNOOP" (Turn off or remove 32K expansion if you have it)
(run #2110)
SNOOP listens for any sound, then returns to basic, statement #2120 will play any message that was left in DIGITIZE MEMORY

If your recorder does not leave mike open with motor off, this routine will not work for you. You could try to use two recorders, one to listen, and one to record, CS1 would have to be on all the time LISTENING, and cs2 used to record any sounds heard.

INSTRUCTIONS

    PLACE SCRATCH TAPE IN RECORDER
    ENGAGE PLAY/RECORD AND PAUSE ON YOUR RECORDER
    THE RED WIRE SHOULD BE PULLED OUT
    ENTER "RUN 2110"
    DISENGAGE PAUSE

    RECORDER SHOULD BE STOPPED AND STARTED WHENEVER A SOUND IS HEARD.

    THESE SOUNDS WILL BE RECORDED ON YOUR TAPE.
    (THIS IS NOT DIGITIZED SOUND WE WILL DO THAT NEXT)

If this routine does not work for you, your recorder probably does not leave mike open when remote control turns off motor, you will probably have to remove all "off1" calls from these examples, and let tape run through your recorder while "RECORD" ing sounds, you will not need tape or recorder to play them back.

SAMPLE PROGRAM #2 "RECORD AND PLAYBACK DIGITALLY"
(run #2450)
This routine digitizes sound and plays it back through sound generator. You have a limited control of sound quality by setting freq of first sound VOICE as shown. Best results will be obtained at frequencies above the audible range.

We are not actually GENERATING sound through the sound generator in a normal sense. We are utilizing the fact that the generator will "FLUTTER" the speaker as we alternate or FLIP/FLOP the signal we are sending.

The delay or TIME between FLIPS creates our frequency reproduction.

To make some comparisons between this program and other SOUND DIGITIZING applications, we can SAMPLE the signal based on frequency only at the RATE of approximately 3000 times per second.

A HI-FI system using digitized sound would sample up to 50,000 times a second and also code additional data.

Texas Instruments Professional Computer with it's voice recognition lab samples signals at approximately 8000 cycles per second.

We have no way to test volume, or other attributes of the sound, we sample only the very simplest concept of frequency, and reproduce that.

Very high frequencies (bell or ting of crystal) will sound very flat, and lower than actual. We probably miss many of the CYCLE changes of the frequencies produced by these sounds.

SAMPLE PROGRAM #3 "STORING PATTERNS LOCALLY IN BASIC ARRAYS"
(run #2710)

If you have succeeded in getting the previous routine to operate, this one should run without any problem.

While the "RECORD" and "PLAY" routines will use the 32K expansion if it is on, the "GET" and "SAY" routines will not work.

If you must leave recorder running to have open mike, remove statement 2750, and try this routine.

"GET" requires an array be passed because basic can only pass 255 byte strings, and our digitized sound will easily exceed that.

Each sound to be saved must be passed a separate array, you can take the data in the array and store it on disk if you wish, or write it to CS2 for later retrieval by using the SEG$ BASIC command to reformat the data to a record size.

SAMPLE PROGRAM #4 "BIO FEEDBACK"
(run #3070)

"GETPAT" is another way to get data from the digitized area.

The string returned will be 32 characters in length.

```
POS 1    will equal the highest value contained in the string.
POS 2-31 will be taken from approximately midway in the currently
         digitized sounds.  (if you enter a constant "OOOOOH" it will be
         fairly representative of that sound)
POS 32   The lowest value in the string.
```

This routine will get the PATTERN string and display a GRAPH of the sound.

Perhaps you can build routines to analyze this data for some primitive sound controlled programs.

REMEMBER:

call link("say") and call link("get") will not work with memory expansion attached. apparently the sound patterns will be routed to the mem-expansion instead of the mini-mem as in call link("record") and call link("play") when used by themselves.

Also, if while experimenting you get erratic results or your computer inadvertently locks up then, well, join the club.

The best way to use these routines is read the docs carefully and look at the examples in the BASIC program that was supplied with them.

I have had no luck at all using the speech data with CALL SAY(SPEECH$) in EXTENDED BASIC. There are control codes in front of each speech string in the ENIE&BERT program and I have not had time to hack them out properly(maybe YOU have the technical scoop on it but I don't). Probably using this data as a "direct-string" rather than a word in CALL SAY will get results but then again maybe not. Though they appear similar, the voice pattern formats may not be compatible in any way, shape, or form.

Unless CALL SAY does work on these strings, I have probably stretched this programs usefulness to the outer limits.

SAVECON will save 5 second or less sound patterns to a file, and OLDCON will play them back one at a time off of the same file; slow but relatively amusing if done right!

I have found that the higher the pitch of the sound or noise digitized, the shorter the recording time, which is no doubt a peculiarity of the digitizing algorithm used.

GOOD LUCK!