Behavior Research Methods, Instruments, & Computers
2001, 33 (1), 65-72

FTAP: A Linux-based program for tapping and music experiments

STEVEN A. FINNEY
Ohio State University, Columbus, Ohio

This paper describes FTAP, a flexible data collection system for tapping and music experiments. FTAP runs on standard PC hardware with the Linux operating system and can process input keystrokes and auditory output with reliable millisecond resolution. It uses standard MIDI devices for input and output and is particularly flexible in the area of auditory feedback manipulation. FTAP can run a wide variety of experiments, including synchronization/continuation tasks (Wing & Kristofferson, 1973), synchronization tasks combined with delayed auditory feedback (Aschersleben & Prinz, 1997), continuation tasks with isolated feedback perturbations (Wing, 1977), and complex alterations of feedback in music performance (Finney, 1997). Such experiments have often been implemented with custom hardware and software systems, but with FTAP they can be specified by a simple ASCII text parameter file. FTAP is available at no cost in source-code form.

Experimental research on finger tapping, motor timing, and music performance has often relied on customized hardware and software configurations, making the replication of such experiments difficult. FTAP, a program written by the author, runs on the Linux operating system and can implement a wide variety of tapping and music performance experiments. It uses standard and generally available MIDI (Musical Instrument Digital Interface) equipment for keystroke input and auditory output and collects and outputs data with verifiable millisecond resolution. The C source code for FTAP, as well as documentation and sample experiment configuration files, is available from the FTAP Web site (http://dactyl.som.ohio-state.edu/ftap).

FTAP runs experiments in which the input data are long sequences of finger movements (keypresses and key releases); a trial may include presentation of a rhythmic auditory stimulus, and control over auditory feedback to the keystrokes may be of interest. FTAP can run many such experiments, including (but not limited to) the following.

1. Synchronization/continuation tapping experiments, including isochronous tapping (Wing & Kristofferson, 1973) and patterned rhythms (Finney, 1999; Finney & Warren, 2000; Vorberg & Hambuch, 1984).
2. Delayed auditory feedback (DAF) experiments in tapping (Chase, Rapin, Gilden, Sutton, & Guilfoyle, 1961; Ruhm & Cooper, 1962) and music (Finney, 1997; Gates, Bradshaw, & Nettleton, 1974), including the combination of synchronous and delayed feedback (Ruhm & Cooper, 1964) and synchronization tasks combined with DAF (Aschersleben & Prinz, 1997).
3. Synchronization tasks with occasional perturbations of the pacing signal (Repp, 2000).
4. Tapping experiments with occasional perturbations to the auditory feedback (Wing, 1977).
5. Auditorily paced multifinger polyrhythmic tapping tasks similar to Jagacinski, Marshburn, Klapp, and Jones (1988) or Krampe, Kliegl, Mayr, Engbert, and Vorberg (2000).
6. Music experiments involving pitch feedback alterations (Finney, 1997).

FTAP is not the first experimental package with the ability to implement some of the above experiments, but it may be the first program that can implement all of them, and it is almost certainly the first freely available open-source distribution to do so.

The basic design of FTAP is shown in Figure 1. MIDI messages are read from an input device (typically, an electronic musical keyboard); these are referred to as *keystroke* events. The data may be transformed (e.g., delayed or altered in pitch) and are then sent to a MIDI output device (a tone generator); these are called *feedback* events. In addition, FTAP can generate its own MIDI output data (e.g., metronome or pacing tones or pregenerated fixed sequences); these are called *metronome* events. These three event types are identified in the output file by the corresponding letters K, F, and M.

FTAP started out as a program implemented on a Silicon Graphics Indigo computer for my own research (Finney, 1997, 1999; Finney & Warren, 2000). I thank Jim Anderson, David Ascher, Peter Eimas, Mike Tarr, and Bill Warren of Brown University for their assistance during this time. The port to Linux and many important enhancements were completed during a postdoctoral fellowship at Ohio State University. I thank Caroline Palmer, Pete Pfordresher, and Shane Ruland for their encouragement and assistance. Portions of this work were supported by NIMH Grant R01-MH45764 to Caroline Palmer. Thanks also go to Rosalee Meyer for comments on an earlier draft of this paper, to David Huron and Doug Reeder for providing a Web home for FTAP (http://dactyl.som.ohio-state.edu/ftap), and to Doug Michels and Paul Iddings for my early UNIX education. Correspondence concerning this article should be addressed to S. A. Finney, Department of Psychology, Ohio State University, 142 Townshend Hall, 1885 Neil Ave., Columbus, OH 43210 (e-mail: sf@dactyl.som.ohio-state.edu).

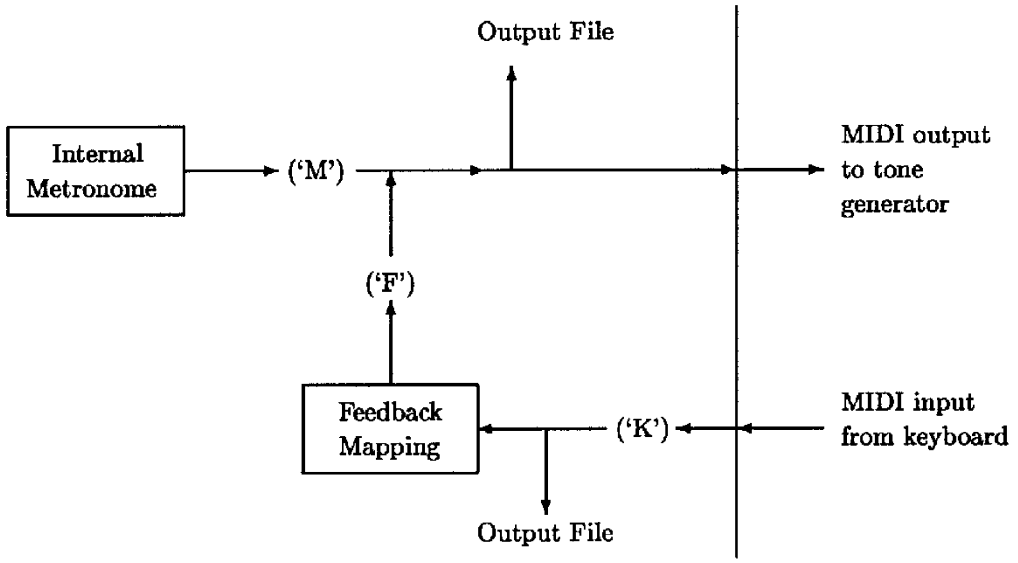

Figure 1. Architecture of FTAP. The letters in parentheses indicate the labeling of a particular event type in the output file; all events go to the same output file.

FTAP is particularly strong in the following areas.

1. Feedback manipulation: Auditory responses to keystrokes can include time delays, pitch alterations, and free combination of delay and pitch alterations.

2. Flexibility: FTAP's behavior can be changed during a trial on the basis of the metronome beat number, keystroke number, or elapsed time. One simple example is stopping the metronome at a specific beat for a synchronization/continuation task; a more complex case is the perturbation of the auditory feedback to one specified keystroke.

3. Sequence generation: A repeating rhythmic metronome or pacing signal (up to a 20-beat cycle) can be programmed. Patterns based on sounded and silent beats, tone length, pitch, and loudness (or any combination of these) are all possible. Perturbations can be specified without the need to prepare a file containing the entire stimulus sequence.

4. Configurability: FTAP experiments are controlled by an ASCII text file that specifies the values for a well-defined set of parameters; this file specifies many aspects of an experiment that would usually be done by low-level programming. FTAP has a total of about 45 parameters, although 20 or fewer suffice for most experiments.

5. Readily analyzable text output: FTAP output files provide millisecond-precision timestamped information for all events (keystroke, metronome, and feedback events) in a columnar text format; this allows easy access to the relationships between the different types of events.

6. Standardized configuration: FTAP runs on standard hardware and software and uses standard MIDI input and output devices.

7. Millisecond resolution: When properly configured, FTAP collects input keystroke data and produces MIDI output with verifiable millisecond precision.

8. User-reproducible performance benchmarking: FTAP's on-line diagnostics notify the experimenter of potential timing problems on each trial. In addition, parameter files are included that allow users to test FTAP's MIDI throughput and timing precision with their own computer and MIDI interface.

Some of FTAP's limitations should also be noted. FTAP is a tool for running a single extended sequential finger movement trial on the basis of a parameter file. FTAP provides no higher level facilities for trial ordering or randomization; this can be done with a shell script or a Python program that handles trial ordering and then calls FTAP. FTAP also does not include any graphical interface. A window-driven experiment design tool could be created that called the FTAP program, but this would be separate from FTAP itself. FTAP's design thus fits the general UNIX/Linux philosophy of writing a tool to do one thing well and then building up a system out of such components. Finally, as will be discussed in more detail below, the actual sounds generated by FTAP depend on MIDI tone generator programming. FTAP provides control over the generation of auditory output only in terms of MIDI note, MIDI velocity, MIDI channel, and the timing of Note On/Note Off messages.

It may be useful to compare FTAP with other tapping and music experiment packages. Todd, Boltz, and Jones (1989) describe MIDILAB, a commercially available system for IBM PCs running DOS. MIDILAB can handle a wide range of experiments and includes high level capabilities, such as trial ordering. However, it is limited in the area of auditory feedback control, and it is not flexible in changing behavior midtrial. Another alternative, the commercial program Max, has been used for tapping and music experiments with great success by Repp (Repp, 1999a, 1999b, 2000; see Winkler, 1998, for extensive dis-

cussion of the Max program). However, Max appears to be available only for Apple Macintosh computers; in addition, since it is not explicitly designed for experimental work, it may have a steep learning curve for psychologists. Up-to-date information on Max is available on line (http://www.cycling74.com). Either MIDILAB or Max might be better suited for the presentation of complex musical stimuli than is FTAP. Mates (1990) describes a program for tapping experiments that has many powerful capabilities, such as changing program behavior on the basis of a real-time analysis of the timing of subjects' responses; the program is available upon request from its author (J. Mates, personal communication, August 2000). However, it is not as flexible as FTAP for multikeyed input and pitched responses, and it requires special hardware modifications (Mates, 1990). Finally, Collyer, Boatright-Horowitz, and Hooper (1997) have used a commercial MIDI sequencer for data collection and stimulus presentation, although this involved a fairly simple experiment without feedback manipulation.

The organization of this paper is as follows. The first section describes some aspects of MIDI that are central to FTAP's implementation. Next, FTAP's parameterized approach to specifying an experiment is described. The subsequent section provides a complete example: a working input file with 13 parameters describing a simple synchronization/continuation experiment. The output file format is then described, followed by a list of some more specialized and advanced capabilities of FTAP. Finally, there is information about computer configuration and the availability of FTAP.

MIDI OVERVIEW

MIDI is a standard protocol for communication between electronic musical instruments and between those instruments and computers; MIDI is compatible with a wide range of commercial keyboards and tone generators. Psychologists' use of MIDI has been documented in more detail elsewhere (Collyer et al., 1997; Kieley, 1991; Todd et al., 1989), so only essential points will be covered here.

The MIDI message protocol includes Note On and Note Off messages (equivalent to keypress and key release events), and each message specifies note (pitch) and key velocity (loudness). Each message also specifies a MIDI channel; this can be useful because a polytimbral tone generator can be programmed to give different sounds on different MIDI channels, and the ability to specify MIDI channels thus allows specification of different sounds. The MIDI data stream itself (as used here) does not provide timing information; FTAP must promptly timestamp input MIDI messages and send output messages at the correct time. The MIDI hardware transmission rate allows a maximum throughput of approximately one message (keystroke event) per millisecond, which is adequate for human performance measurement.

Using MIDI for data input allows for a range of input devices. MIDI keyboards are readily available, fairly cheap, and allow simultaneous data collection from multiple fingers; they also usually provide a form of keystroke force information in the form of MIDI velocity. However, MIDI keyboards also have some disadvantages: A half inch of finger movement may be required to signal a keystroke, and there is no measurement in standard units for keystroke force or velocity (i.e., MIDI velocity values can be assumed to be monotonically increasing with keystroke velocity, but they do not directly translate to any other type of velocity or force unit, nor are they comparable across different keyboards). Other devices using a MIDI interface could be used in place of a keyboard—for example, an electronic drum pad or a custom MIDI device.

For output, FTAP controls the timing of MIDI messages but does not directly control the sound. That is, FTAP will send precisely timed MIDI Note On and Note Off messages to a tone generator (with the MIDI note, velocity, and channel specified in each message), but the resulting sound characteristics depend on the tone generator. For example, MIDI velocity controls loudness, but the relation of MIDI velocity to sound level depends on the specific tone generator, chosen timbre, and the amplification equipment. The tone generator also determines the attack and offset characteristics of the sound. FTAP may transmit a MIDI Note Off message 30 msec after a Note On message, but that alone does not guarantee the generation of a clean 30-msec duration tone. Determining the suitability of a particular voice or timbre on a particular tone generator is the user's responsibility.

BASIC PARAMETERS

FTAP runs a single (possibly long) trial, based on a set of parameters listed in an ASCII text file. A parameter describes a characteristic of the experimental situation, such as whether there is feedback to the subject's keystroke, whether the feedback is delayed, the time between beats of the metronome, and so forth. Each parameter controls one specific aspect of the experiment; in general, parameters may be flexibly combined in interesting ways. Any parameters not explicitly specified in the input file default to a documented and rational inactive value. The most important parameters have an integer value ("integer parameters"), although there are also parameters with string and array values.

Metronome Parameters

FTAP can provide a metronome (or pacing) signal for synchronization experiments; this can also serve as a metronome for musical performance experiments. In the simplest case, a metronome tone has a fixed pitch, velocity, and duration and occurs at a constant interstimulus interval (notated as MSPB: milliseconds per beat). The following parameters specify a metronome rate of two beats per second (MSPB = 500), with a tone on MIDI channel 1 (MET_CHAN = 1) sounding for 30 msec (MET_LEN = 30), and with fixed MIDI velocity of 90 (MET_VAL = 90) and MIDI note number of 86 (MET_NOTE = 86, equiva-

lent to the musical pitch D6). The METRON_ON value of 1 means the metronome will sound; a value of 0 would cause it to be silent.

```
MSPB        500
MET_CHAN    1
MET_LEN     30
MET_VEL     90
MET_NOTE    86
METRON_ON   1
```

It is possible to impose a pattern on the metronome. The parameter specification below (an *array* parameter, specified as a list length followed by the list elements) will configure the metronome as a four-beat pattern of three sounded beats and a pause (1 indicates a sounded beat, 0 a silent beat).

```
MET_PATTERN_ARRAY    4    1 1 1 0
```

The following array specification would cause the metronome to have a louder sound (higher MIDI velocity) on the first beat of each pair of beats:

```
MET_VEL_ARRAY        2    110  90
```

Pitch, tone length, and MIDI channel can be similarly specified; the current maximum pattern length is 20 beats.

Feedback Parameters

FTAP provides a high degree of control over the auditory feedback resulting from a subject's keystroke. At any point in time, feedback may be on or off (the FEED_ON parameter set to 1 or 0, respectively). The output pitch may be the same as the input keystroke, or it can be set to a fixed value or altered from the input keystroke value in a number of ways. The feedback loudness (MIDI velocity) may be determined on the basis of input keystroke velocity or may have a fixed value. The timing of feedback may be synchronous with the keystroke or delayed by any amount.

In a simple case, the keyboard input directly controls the auditory feedback, as in normal musical keyboard performance. The parameters below specify that the auditory output should be identical to the subject's keystrokes, although it is forced to MIDI channel 1. The value of 0 for FEED_PMODE (pitch mode), FEED_VMODE (velocity mode), and FEED_LEN indicates that the input keystroke values (those produced by the subject) for MIDI note, velocity, and tone length should be used.

```
FEED_CHAN    1
FEED_PMODE   0
FEED_VMODE   0
FEED_LEN     0
```

In other cases (e.g., a tapping experiment), a feedback tone of fixed pitch, loudness, and duration might be preferable. The following parameters will send out such a tone, with fixed pitch (MIDI note 76, or E5), fixed loudness (MIDI velocity 90), and fixed length (100 msec), regardless of the characteristics of the subject's keystroke. Setting the PMODE and VMODE parameters to 1 specifies a fixed output value.

```
FEED_PMODE   1
FEED_NOTE    76
FEED_VMODE   1
FEED_VEL     90
FEED_LEN     100
```

Perhaps more interesting are the delay and altered pitch mappings. The FEED_DMODE (delay mode) parameter controls the delay feedback mode. A value of 0 will give synchronous feedback, whereas a value of 1 will delay feedback for the keystroke by a fixed amount (determined by the FEED_DVAL parameter). The following parameters would delay auditory feedback to a subject's keystrokes by 250 msec.

```
FEED_DMODE   1
FEED_DVAL    250
```

The pitches that occur in response to keystrokes are controlled by the FEED_PMODE parameter. There are parameter values that allow for quasirandom pitches, for logically reversing the keyboard (putting high notes on the right), or for playing notes from a prespecified sequence. Such pitch manipulations are useful for music experiments (e.g., Finney, 1997), and musicians tend to find them amusing. They may also be useful for experiments in sequence learning.

Pitch alterations and delay can be freely combined by setting FEED_PMODE and FEED_DMODE appropriately. For example, setting FEED_DMODE to 1 and FEED_PMODE to 4 would give random pitch output that is also delayed.

Trigger Events

An important part of FTAP's flexibility is the ability to define *trigger events*. Trigger events change the value of an integer parameter in midtrial on the basis of metronome count, keystroke number, or elapsed time; this allows alteration of behavior during the course of a trial. One simple example occurs with the standard continuation paradigm, in which the metronome sounds for a specified number of beats and then stops, while the subject continues tapping. A more complex case is exemplified by Wing (1977), in which the auditory feedback for a single keystroke was delayed in the midst of otherwise uniformly timed feedback.

In the parameter file, triggers are specified by an initial field containing the keyword TRIGGER. The second field is a unique trigger ID (for identification in the output file), and the third field is the trigger type: K(eystroke), M(etronome), or T(ime). The fourth field is the count for the trigger (keystroke number, metronome count, or milliseconds since trial start), and the next two fields are the parameter to change and the new value. The following parameters would turn the metronome off after 15 beats (e.g., for a continuation paradigm) while simultanously turning on feedback to the subject's keystrokes. Here, multiple parameter changes occur in response to a single metronome event.

```
TRIGGER 1 M  16 METRON_ON   0
TRIGGER 2 M  16 FEED_ON     1
```

A special pseudoparameter END_EXP terminates a trial. The first line below would terminate the trial on the subject's 56th keystroke; the second would end the trial after 30 sec.

```
TRIGGER 1 K  56     END_EXP 0
TRIGGER 2 T  30000  END_EXP 0
```

The next pair of events (with FEED_DMODE = 1 and FEED_ON = 1) would cause feedback to the subject's 40th keystroke to have a delay of 75 msec, whereas later keystrokes would have a feedback delay of 40 msec (as in Wing, 1977).

```
TRIGGER 1 K 40 FEED_DVAL    75
TRIGGER 2 K 41 FEED_DVAL    40
```

SAMPLE TRIAL

A complete parameter file for a simple synchronization experiment is shown in Figure 2. Using this file, FTAP would generate a pacing tone 30 msec long, with fixed loudness (fixed MIDI velocity of 100) and fixed pitch (C6, or MIDI note 84); the tones will be generated every 250 msec. A subject's keystroke will produce auditory feedback with fixed pitch (E4, or MIDI note 64), fixed loudness (MIDI velocity 90), and fixed duration (100 msec). If this text were in a file named "Cont250," the trial could be run by the following command:

```
ftap Cont250
```

Minor modifications to the file in Figure 2 would change the characteristics of the trial. For example, altering the values for MET_VEL or FEED_VEL would change the loudness of the pacing signal or the feedback, respectively. Changing the value for MSPB would change the rate of the pacing signal. A trigger could be added to set FEED_ON to 0 at metronome beat 16, which would change the experiment so that auditory feedback would be on only during the synchronization phase. The auditory feedback to keystrokes could be delayed by setting FEED_DMODE to 1 and setting FEED_DVAL to the desired delay value. The period of the synchronization signal could be increased by 10 msec in midexperiment by adding a trigger changing MSPB to 260.

OUTPUT FILE FORMAT

The output file records keystroke, feedback, and metronome events. It also records MIDI controller information, the starting parameters for the trial, the trigger events occurring during the trial, and some internal performance diagnostics. The file is columnar in format; a sample of

```
# The trial starts with an isochronous pacing signal, with an ISI of 250
# milliseconds, and a 30 millisecond tone of fixed pitch and loudness.
# The VEL, NOTE, and LEN parameters must be set to reasonable values for
# the tone generator being used.

METRON_ON       1
MSPB            250
MET_VEL         100
MET_NOTE        84
MET_LEN         30

# Feedback to the subject's keystrokes is on at the start of the trial;
# the  resulting tones are of fixed loudness (MIDI velocity value of 90),
# fixed pitch (MIDI note 64) and of fixed 100 ms length.

FEED_ON         1
FEED_VMODE      1
FEED_VEL        90
FEED_PMODE      1
FEED_NOTE       64
FEED_LEN        100

# The pacing signal (metronome) stops sounding at the 16th metronome beat,
# making this a synchronization/continuation experiment.

TRIGGER 1 M 16     METRON_ON 0

# End the experiment after 20 seconds (20000 milliseconds).

TRIGGER 2 T 20000  END_EXP   0
```

Figure 2. Sample FTAP parameter file for a synchronization/continuation tapping experiment with an isochronous 250-msec interstimulus interval pacing signal. All output defaults to MIDI channel 1. "#" indicates a comment line.

Table 1
FTAP Output File for a Synchronization Experiment (Excerpt)

Time (msec)	Up/Down	MIDI Channel	MIDI Note	Note Name	MIDI Velocity	Sequence Number	Type
1000	D	1	84	C6	100	0	M
1030	U	1	84	C6	0	0	M
1223	D	1	60	C4	112	3	K
1224	D	1	64	E4	80	3	F
1250	D	1	84	C6	100	0	M
1280	U	1	84	C6	0	0	M
1323	U	1	64	E4	0	0	F
1330	U	1	60	C4	0	0	K
1467	D	1	60	C4	101	4	K
1467	D	1	64	E4	80	4	F
1500	D	1	84	C6	100	0	M
1530	U	1	84	C6	0	0	M
1567	U	1	64	E4	0	0	F
1577	U	1	60	C4	0	0	K

data lines (covering 600 msec) including keystroke, feedback, and metronome events is shown in Table 1. This is a portion of an output file from a subject in the experiment specified by the parameter file in Figure 2.

The first field is the millisecond time of the event, relative to trial start. The second field is "D" or "U," depending on whether the event is a key down or key up (i.e., MIDI Note On or Note Off). The third field is the MIDI channel. The fourth field is the MIDI note value (an integer), and the fifth field is the pitch name representation of that note. The sixth field is the MIDI velocity, and the seventh is a sequence number for input keystrokes. The eighth field is particularly important and defines the type of event: "K" for subject keystrokes, "M" for metronome events, and "F" for feedback events. This field can be used to restrict analysis to particular event types. For instance, if the only data of interest are keystroke events, the relevant data lines can be extracted on the basis of the value of "K" in this column.

Some interesting features can be observed in Table 1. All the events are strictly ordered in time, regardless of the event type; this allows looking at temporal relationships between any types of events. For instance, the characteristic anticipation found in synchronization tapping is demonstrated by the keystroke down event at time 1223, which precedes the corresponding metronome beat at time 1250 by about 30 msec. Feedback to keypresses is synchronous with the down keystroke, although there may be an occasional millisecond difference owing to processing overhead and roundoff error (see times 1223 and 1224). Keystroke release "feedback" appears, somewhat counterintuitively, to precede the associated keystroke release (e.g., the events at times 1323 and 1330), but recall that in this experiment, feedback was defined as a fixed length 100-msec tone. That is, Note Off "feedback" events are independent of the key release. Finally, even though the sound the subject hears is of fixed loudness (the MIDI velocity for feedback events is always 80), the varying velocities of the keystrokes themselves are still recorded.

ADVANCED FEATURES

FTAP has a number of other capabilities; some of these are primarily for tapping experiments (i.e., involving a single finger on a single key), whereas others are primarily for music experiments.

1. Random delays: In addition to fixed delay values, it is possible to have the delay for each keystroke randomly selected from a list of delays or selected from a uniform distribution (which is currently hard-coded to the range of 100–300 msec).

2. Velocity alterations: In addition to the choice of fixed or variable velocity (loudness) for auditory feedback, there are also some preliminary velocity mappings (e.g., harder keystrokes can be made to cause softer sounds).

3. Multiple feedback channels: FTAP provides a second logical "feedback channel," which permits either two independent feedback responses to one keystroke (e.g., combining synchronous and delayed feedback) or different types of feedback response for two parts of the keyboard (defined by a split point).

4. Fixed sequences: FTAP can play a preprogrammed stimulus (a fixed sequence of notes) from a file.

5. Masking noise: A tone can be programmed to go on at the beginning of the experiment and off at the end of the experiment; this can be used to provide masking noise throughout the experiment when used in conjunction with a suitable tone generator timbre.

6. Polyrhythms: The ability to have a metronome pattern that is up to 20 beats long, with control over whether each beat is played (as well as pitch, length, and loudness for each beat), allows for a limited range of polyrhythmic pacing signals for use with either synchronization or continuation tasks.

7. MIDI controllers: FTAP can record most MIDI controller messages (e.g., sustain pedal or pitch bend); these are listed in the output file as event type "C." Such controller messages do not undergo complex feedback mapping.

8. User customization: Although there is not currently an explicit interface for adding user-specified mappings for pitch or delay, it is relatively simple to do so with the provided source code.

REAL-TIME DIAGNOSTICS AND BENCHMARKING

FTAP provides precise millisecond-resolution data collection on the Linux operating system; this claim may surprise those who are aware of the potential problems of real-time data collection on a complex multiuser, multitasking operating system. However, the combination of modern fast hardware (e.g., 300-MHz Pentium processor

based computers with large amounts of RAM), real-time support provided by standard Linux, and careful coding makes such precision possible. More detailed information on performance, benchmarking, and implementation is available in Finney (2000) and the FTAP Reference Manual; the interested reader can also consult the provided C code. Running FTAP with reliable millisecond timing uses real-time features of Linux that require root privileges. Configuration of FTAP as a *setuid root* program is recommended for serious data collection, although testing and evaluation can be done with normal user privileges.

Users can verify FTAP's performance for themselves on their own systems. One particular issue is verifying the timing capabilities of the user's own MIDI hardware and drivers. A parameter file included with the distribution can be used in conjunction with a MIDI loop configuration (using a cable to connect the MIDI output back to the MIDI input) to test the maximal throughput of FTAP. This parameter file is configured to provide feedback to keystrokes and also sends out a single metronome event. This output MIDI message is immediately interpreted as keystroke input (because the MIDI output is connected to the input) and then generates a MIDI output message as feedback. The single message will continuously loop through the system, providing a strong test of throughput. The performance can be checked by inspection of the generated output file, as well as by comparing against elapsed time. Such testing shows that FTAP, on a properly configured system, attains the maximal bandwidth possible under MIDI (approximately one MIDI event per millisecond; see Table 2).

In addition, the internal timing of each run of FTAP (i.e., the detection of possible operating system scheduling problems) is clearly indicated in output diagnostics printed both to the screen and to the output file. With a properly configured system, serious problems do not occur. For example, in a tapping experiment with a total of 1,320 thirty-second trials (Finney & Warren, 2000), only 50 trials (3.8%) showed a scheduling discrepancy of greater than 1 msec. These 50 trials each contained a single scheduling discrepancy of either 2 or 3 msec, and these small discrepancies typically occurred when no MIDI data were being processed. This seems quite acceptable for more than 10 h of data collection.

Table 2
Sample FTAP Output From the Loop Benchmark Test

Time (msec)	Up/Down	MIDI Channel	MIDI Note	Note Name	MIDI Velocity	Sequence Number	Type
8	D	1	86	D6	100	3	K
8	D	1	86	D6	100	3	F
9	U	1	86	D6	0	0	K
9	D	1	86	D6	100	4	K
9	U	1	86	D6	0	0	F
9	D	1	86	D6	100	4	F
10	U	1	86	D6	0	0	K
10	U	1	86	D6	0	0	F
11	D	1	86	D6	100	5	K
11	D	1	86	D6	100	5	F
12	U	1	86	D6	0	0	K
12	U	1	86	D6	0	0	F
13	D	1	86	D6	100	6	K
13	D	1	86	D6	100	6	F
14	U	1	86	D6	0	0	K
14	U	1	86	D6	0	0	F

Note—Both input and output are processed with millisecond resolution.

CONFIGURATION ISSUES

FTAP has been run by the author primarily on a 200-MHz Pentium computer running a RedHat 6.2 Linux distribution (a 2.2 kernel), with a Creative SoundBlaster 16 card as the MIDI interface and an external tone generator (a Yamaha TX81Z) for auditory output. MIDI access is done via the generic device "/dev/midi" so there is no dependence on a particular sound card, although not all sound card/driver combinations can handle the heavy throughput of the loop benchmark described above (see the FTAP Reference Manual for details). Experiments should be run on a machine dedicated to data collection (e.g., not running as a network server or running other users).

One particular concern is that the OSS-Free MIDI drivers that are currently shipped with many Linux distributions process MIDI output with a 10-msec poll (at least with MIDI hardware interfaces that do not have interrupt capability); this does not give the desired millisecond control of output. The problem can be readily demonstrated by the loop benchmark described above. Although such output timing granularity may not be critical in some experimental situations, it might be a serious problem in others. One solution is to use the low-cost, binary-only OSS drivers from 4Front Technologies (http://www.4front-tech.com); these drivers do not suffer from the 10-msec polling problem. Alternatively, the open-source Advanced Linux Sound Architecture drivers (http://www.alsa-project.org) are also claimed to be free of this problem.

AVAILABILITY

FTAP is available at no cost (but without warranty) from the FTAP home page (http://dactyl.som.ohio-state.edu/ftap) or by contacting the author (sf@dactyl.som.ohio-state.edu). The distribution includes compileable C source code and binaries for a Pentium Linux system; the code is covered by the GNU Public License agreement. Sample parameter files are also included, with demonstrations of various FTAP features and sample experiments (such as those listed in the introduction). The distribution also includes a user's guide and a reference manual; this documentation is currently somewhat rough but should be accurate and complete.

REFERENCES

Aschersleben, G., & Prinz, W. (1997). Delayed auditory feedback in synchronization. *Journal of Motor Behavior*, **29**, 35-46.

Chase, R., Rapin, I., Gilden, L., Sutton, S., & Guilfoyle, G. (1961). Studies on sensory feedback: II. Sensory feedback influences on key-tapping motor tasks. *Quarterly Journal of Experimental Psychology*, **13**, 153-167.

Collyer, C. E., Boatright-Horowitz, S. S., & Hooper, S. (1997). A motor timing experiment implemented using a musical instrument digital interface (MIDI) approach. *Behavior Research Methods, Instruments, & Computers*, **29**, 346-352.

Finney, S. A. (1997). Auditory feedback and musical keyboard performance. *Music Perception*, **15**, 153-174.

Finney, S. A. (1999). *Disruptive effects of delayed auditory feedback on motor sequencing*. Unpublished doctoral dissertation, Brown University.

Finney, S. A. (2000). *Real-time data collection in Linux: A case study*. Manuscript submitted for publication.

Finney, S. A., & Warren, W. H. (2000). *Delayed auditory feedback and rhythmic tapping: Evidence for a critical interval shift*. Manuscript submitted for publication.

Gates, A., Bradshaw, J. L., & Nettleton, N. C. (1974). Effect of different delayed auditory feedback intervals on a music performance task. *Perception & Psychophysics*, **15**, 21-25.

Jagacinski, R. J., Marshburn, E., Klapp, S. T., & Jones, M. R. (1988). Tests of parallel versus integrated structure in polyrhythmic tapping. *Journal of Motor Behavior*, **20**, 416-442.

Kieley, J. M. (1991). MIDI and Macintosh: Searching for a better mousetrap. *Behavior Research Methods, Instruments, & Computers*, **23**, 256-264.

Krampe, R. T., Kliegl, R., Mayr, U., Engbert, R., & Vorberg, D. (2000). The fast and the slow of skilled bimanual rhythm production: Parallel versus integrated timing. *Journal of Experimental Psychology: Human Perception Perception & Performance*, **26**, 206-233.

Mates, J. (1990). A system of personal computer control programs for tapping experiments. *Computer Methods & Programs in Biomedicine*, **33**, 43-48.

Repp, B. H. (1999a). Control of expressive and metronomic timing in pianists. *Journal of Motor Behavior*, **31**, 145-164.

Repp, B. H. (1999b). Detecting deviations from metronomic timing in music: Effects of perceptual structure on the mental timekeeper. *Perception & Psychophysics*, **61**, 529-548.

Repp, B. H. (2000). Compensation for subliminal timing perturbations in perceptual–motor synchronization. *Psychological Research*, **63**, 106-128.

Ruhm, H., & Cooper, W. (1962). Low sensation level effects of pure-tone delayed auditory feedback. *Journal of Speech & Hearing Research*, **5**, 185-193.

Ruhm, H., & Cooper, W. (1964). Influence on motor performance of simultaneous delayed and synchronous pure-tone auditory feedback. *Journal of Speech & Hearing Research*, **7**, 175-182.

Todd, R. E., Boltz, M., & Jones, M. R. (1989). The MIDILAB music research system. *Psychomusicology*, **8**, 83-96.

Vorberg, D., & Hambuch, R. (1984). Timing of two-handed rhythmic performance. In J. Gibbon & L. Allan (Eds.), *Timing and time perception* (Annals of the New York Academy of Sciences, Vol. 423, pp. 390-406). New York: New York Academy of Sciences.

Wing, A. M. (1977). Perturbations of auditory feedback delay and the timing of movement. *Journal of Experimental Psychology: Human Perception & Performance*, **3**, 175-186.

Wing, A. M., & Kristofferson, A. B. (1973). The timing of interresponse intervals. *Perception & Psychophysics*, **13**, 455-460.

Winkler, T. (1998). *Composing interactive music: Techniques and ideas using Max*. Cambridge, MA: MIT Press.

(Manuscript received August 27, 2000;
accepted for publication December 17, 2000.)