SOFTWARE PERFORMANCE AND TIMING ANALYZER

20665 FOURTH STREET * SARATOGA, CA 95070 * (408) 741-5900

May 9, 1986

TABLE OF CONTENTS

CHAPTER 1. SPTA OVERVIEW

Section **Page**

CHAPTER 2. GETTING STARTED WITH SPTA

Section **Page**

CHAPTER 3. USING SPTA COMMANDS

Section **Page**

CHAPTER 4. SPTA COMMAND REFERENCE

APPENDICES

FIGURES

TABLES

CHAPTER 1. SPTA OVERVIEW

INTRODUCTION

This manual covers the Software Performance & Timing Analyzer (SPTA) that is used in conjunction with the PC PROBE and the stand-alone version of the SPTA. This manual was written as a supplement to the standard PROBE manual and should be inserted at the end of that manual.

FILES ON YOUR SPTA DISKETTE

There are several files on your SPTA diskette which may or may not be needed depending upon what you are doing. A list of these files and their purpose is given in Appendix B. You should make a working copy of these files, and save the original disk for a backup.

MEASUREMENT TOOLS

The SPTA commands include a set of simple timing measurement and performance measurement tools. These can be used to determine the relative performance of applications program modules, time program execution of specific modules, or count program events. An overview of these measurement tools is given next. A more detailed description of each measurement tool is given in Chapter 4, SPTA COMMAND REFERENCE.

PROGRAM ACTIVITY MEASUREMENT

The Program Activity measurement (PA measurement) does a statistical analysis of where a program is spending its time. The CS:IP is sampled at a rate you specify and is stored in PROBE memory. The measurement can be terminated by reaching the maximum number of samples, pressing the STOP button, or reaching a specified stop address. The choice of sample rate depends on the duration of the program being measured. A program taking several minutes to execute should have a longer sampling time than one executing in a few seconds. If the sample rate is too short with respect to the total execution time of the program to be measured, then the measurement will terminate because the sample space is filled. See Table C-1 in Appendix C for additional information.

The result of the PA measurement is a histogram display of the time spent in user defined address ranges. These address ranges are set with the Arange command. Since the data for a particular measurement is stored in the PROBE memory space, a given measurement can display a histogram over different ranges of memory by simply changing the ranges with the Arange command and redisplaying a new histogram. The CS:IP sample taken by the PA measurement is not transparent to the program under measurement and will slow the program down. The amount of time "stolen" from the program for each measurement is shown in Appendix C. For program dead spots such as waiting for keyboard or disk IO, the PA measurement allows regions of the program to be excluded from being entered as samples in the measurement. Exclude regions allow program wait loops to not impact the desired measurement with unimportant data.

PROCEDURE DURATION MEASUREMENT

The Procedure Duration measurement (PD measurement) measures the duration time of a specified procedure, which may have multiple entry and exit points. A timer on the PROBE is turned on at each entry point and turned off at each exit point. The measurement can be terminated by reaching the maximum number of samples, pressing the STOP button, or reaching a specified stop address. A histogram display of the execution time for the procedure is displayed over user specified time ranges. The time ranges are specified by the Trange command. Since the time samples are stored in PROBE memory, new histograms may be displayed over different time ranges by using the Trange command. The PD measurement is useful for finding the minimum, average, and worst case execution times for a given procedure. The time measurements taken by the PD measurement are not transparent to the program under measurement and will slow the program down. The amount of time "stolen" from the program for each measurement is shown in Appendix C. The amount of time used by the PD measurement is not included in the measured sample itself, and the measured time is accurate to 0.1 millisecond.

PROGRAM EVENT COUNTER

The Program Event counter (PE counter) enables you to specify up to 16 executable instructions as events. These events are counted and a histogram is displayed. The histogram shows the number of counts of each event as a % of all other events in the measurement. The measurement is terminated by reaching the maximum number of samples, pressing the STOP button, or reaching a specified stop address. The PE counter is useful for counting the number of iterations through complex nested loops. It can count the number of times different procedures are called, and therefore it can optimize an overlay structure for a program. The event measurements by the PE counter are not transparent to the program under measurement and will slow the program down. The amount of time "stolen" from the program for each measurement is shown in Appendix C.

PROCEDURE TIMING ANALYSIS

The Procedure Timing Analysis (PTA) stores the PROBE timer value each time a user specified event (executed instruction) is detected. A linear display of the sequence of events and the relative time between these events is provided. The measurement is terminated by pressing the STOP button, reaching a specified stop address, or storing the maximum number of timer readings for a given program. The time measurements taken by the PTA command are not transparent to the program under measurement and will slow the program down. The amount of time "stolen" from the program for each measurement is shown in Appendix C. The amount of time used by the PTA is not included in the measured sample itself, and the measured time is accurate to 0.1 milliseconds.

HISTOGRAM DISPLAY FORMAT

A histogram is a graphical display of accumulated data. The SPTA measurement display format is shown below.

```
Type of measurement


Range Min          Range Max          Count  %  0   20  40  60  80  100

Range0
Range1
Range2
Range3
Range4
Range5
Range6
Range7
Range8
Range9
RangeA
RangeB
RangeC
RangeD
RangeE
RangeF
                                      Total:
```

The vertical axis entries are your defined ranges. The ranges for a measurement may be address ranges, time ranges, or events. The horizontal axis is %. The histogram shows the number of occurrences in each range as a % of the sum total of all occurrences in all samples collected. The number of occurrences in the range is also shown in the count column.

After a measurement is taken, a different set of ranges may be specified by simply redefining the current range numbers, and a new histogram can be calculated and displayed. This allows a single measurement to be viewed over a different spectrum of ranges.

Standard default ranges enable you to get a "handle" on the measurement when no previous starting point is available. Address ranges may be specified symbolically using symbols in the symbol table. A total of 16 discrete ranges may be specified for each measurement. The specified ranges may not include all samples taken in the measurement. If the TOTAL does not equal 100%, then there are additional samples which do not appear in the displayed histogram.

CHAPTER 2
GETTING STARTED WITH SPTA

STARTING UP SPTA

To start the SPTA enter:

PERF [/PATH SPEC]	For the PC PROBE version
SWPERF [/PATH SPEC]	For the stand-alone version

[/path spec] is the path for the additional SPTA overlay file if it is not in the current directory. If **[/path spec]** is not included in the command line and these files cannot be found in the current system directory, then SPTA queries the user for the current path.

EXAMPLE: A:\ATRON\ATPERF /A:\ATRON\

This command loads the SPTA software from directory ATRON on drive A even if this is not the current drive and directory.

SPTA COMMANDS

The SPTA commands include both debugging commands and performance measurement commands. This combination lets you debug a program as well as check its performance analysis. The SPTA commands which cause a measurement to be taken require more than one line of input, and the command prompts you for the next line of parameters. A summary of SPTA commands is given in Chapter 4, SPTA COMMAND REFERENCE.

MENU OF COMMANDS

Your interface to the SPTA is similar to the PROBE and features the optional menu of commands at the bottom of the screen. The command set for the SPTA is shown in the menu windows below. If you have the stand-alone version of the SPTA, then the actual menu windows will differ from those shown below.

MENU 1

```
Arange BP BYte COMpare CONsole DElete ECho EMacro EVal FIll FLag Go IF INit
INTerrupt LIst LOAd LOOp MAcro MEnu MODule MOVe NEst NOVerify PAm MOre
```

MENU 2

**PDm PEc POrt PRint PTA PTR Quit Register SAve SCreen SEArch SELect
STep SYmbol Trange Unasm WIndow WOrd MORe**

CONTROLLING OUTPUT

The ESC typed at any time will cancel the current command line. Commands which output large amounts of data to the screen can be stopped with with the Ctrl Break (or Ctrl C) key or the STOP button. Ctrl Break or the STOP button will let you terminate data entry into the PA measurement, PD measurement, PTA, and PE counter commands.

CHAPTER 3
USING SPTA COMMANDS

A SAMPLE MEASUREMENT SESSION

This chapter provides an overview on using the SOFTWARE PERFORMANCE AND TIMING ANALYZER (SPTA) by using an example to demonstrate a typical measurement scenario. The program to be measured is a Pascal program and the source, object, and map files for the program are included on the SPTA disk to allow you to actually try the example in real time.

The example program demonstrates measuring a program which is interactive with the operating system, keyboard, and display. This program does a memory test on system memory. It also prompts you to input the starting boundary of the K of memory to be tested and the number of K bytes to be tested. A summary of the program is shown below.

DEMO PROGRAM LISTING

```
************************************************************
{$Title: 'Memory Test Program'}
{$Subtitle: 'Main Module Declarations'}
{$Linesize: 100}
{$Pagesize: 60}
{$Debug-,Indexck-,Mathck-,Nilck-,Rangeck-,Stackck-}

program Memory_Tester (input, output);

Const
OneK      = 1024;  {1024 bytes in 1K}
OneKPara  = 64;    {64 paragraphs in 1K}

TestValue = #A5;

VAR
StartingK [public]: word;
NumberK   [public]: word;

TestedK    [public]: word;
NumberRead  [public]: word;

MemAddr   [public]: Ads of byte;

```

```
          {$Page+}
          {$Subtitle: 'Initialize'}

          procedure Initialize [public];
          {-----------------------------------------------
           Prompt the user for the starting K of memory to test
           and the number of K to test.
          -----------------------------------------------}

          begin
          {-----------------------------------------------
          First get the starting K.  Note that the
          address space (1Mb) is made up of 1024 1K blocks.
          -----------------------------------------------}
          repeat
             writeln;
             write ('What K would you like to start testing (0 -');
             write ((OneK - 1):4, ')? ');
             readln (StartingK)
          until (StartingK < OneK);

          {-----------------------------------------------
          Now get the number of K.  Make sure StartingK +
          NumberK does not go over the end of memory.
          -----------------------------------------------}
          repeat
             write ('How many K would you like to test (1 -');
             write ((OneK - StartingK):4, ')? ');
             readln (NumberK);
          until (NumberK <> 0) and ((StartingK + NumberK) <=
          OneK);

  {-----------------------------------------------
  Now set MemAddr.S and MemAddr.R for start of test.
  -----------------------------------------------}
  MemAddr.S := StartingK * OneKPara;
  MemAddr.R := 0;
end;
```

```
    Procedure TestMemory [public];
    {-----------------------------------------------
    Start at MemAddr and test 1K blocks of memory for
    Numberk blocks.
    -----------------------------------------------}

begin
{-------------------------------------------
Test blocks OneK at a time.
-------------------------------------------}
TestedK := 0;
while (TestedK < NumberK) do begin
  {-------------------------------------------
  Test the OneK block.
  -------------------------------------------}
  while (MemAddr.R < OneK) do begin
    MemAddr^   := TestValue;
    NumberRead := MemAddr^;

    if (NumberRead <> TestValue) then begin
      {-------------------------------------
      Error found.  Report it.
      -------------------------------------}
      writeln ('Error found at ',
        MemAddr.S:4:16, ':', MemAddr.R:4:16);
    end;

    MemAddr.R := MemAddr.R + 1;
  end;
  {-------------------------------------------
  Move up to next K.
  -------------------------------------------}
  MemAddr.S := MemAddr.S + OneKPara;
   MemAddr.R := 0;

   TestedK := TestedK + 1;
 end;
 writeln;
 writeln ('Test completed.');
 end;
```

```
{$Subtitle: 'Main Program'}
{-------------- Main Program -------------------}
begin
   Initialize;
   TestMemory;
end.
```

EXERCISING THE DEMO

In the examples below, the typed input is shown in bold. First invoke the SPTA software.

PERF <enter>	For the PC PROBE version
SWPERF <enter>	For the stand-alone version

Next load a macro file called PERF.MAC which has been previously created and will prove useful for many different timing analysis sessions.

-LOA M PERF.MAC <enter>

Now take a look at the names of the macros from this file which are now loaded into the PROBE macro table.

```
-MAC <enter>

INIT
DELARALL
DELTRALL
MEMTESTADDR
64K
64K2
4K
4K2
1K
1K2
256B
256B2
16B
16B2
1B
1B2
US
US2
MS
MS2
10MS
10MS2
100MS
100MS2
SEC
SEC2
```

Execute the init macro and pass the file to be loaded (MEMTEST). As this macro executes the screen will look as follows:

```
-EM INIT MEMTEST <enter>

-LOA MEMTEST.EXE
-LOA S MEMTEST.MP1
READING SYMBOLS
READING LINES.
```

Since this macro has loaded the symbol table, symbols can now be displayed. Note that this symbol table has been previously stripped of symbols in the Pascal run-time library and only user program symbols remain.

-SY <enter>

```
Address          Symbol name
09BF:0001        INITIALIZE
0D60:F5B0        MEMADDR
09BF:01FC        MEMORY__TESTER
0D60:F5AA        NUMBERK
0D60:F5AE        NUMBERREAD
0D60:F5A8        STARTINGK
0D60:F5AC        TESTEDK
09BF:011A        TESTMEMORY

Line numbers for module ..MEMORY__TESTER
#38=09BF:0008    #39=09BF:0008    #40=09BF:0011    #41=09BF:0028
#42=09BF:0054    #49=09BF:007A    #50=09BF:007A    #51=09BF:0091
#52=09BF:00C1    #53=09BF:00DF    #58=09BF:0105    #59=09BF:0110
#60=09BF:0116    #75=09BF:0121    #76=09BF:0127    #80=09BF:0133
#82=09BF:013E    #83=09BF:0146    #85=09BF:0152    #89=09BF:015A
#91=09BF:01B6    #93=09BF:01B6    #94=09BF:01BA    #99=09BF:01BD
#100=09BF:01C2   #102=09BF:01C8   #103=09BF:01CC   #105=09BF:01CF
#106=09BF:01D8   #107=09BF:01F8   #113=09BF:0208   #114=09BF:020D
#115=09BF:0212
```

NOTE: The segment values may vary with memory size and OS.

Inspect the macro MEMTESTADDR which will set the address ranges for the first Program Activity measurement.

```
-MAC MEMTESTADDR <enter>
```

```
EM DELARALL
A 0=0:0 .initialize
A 1=#75 #80-1
A 2=#80 #85-1
A 3=#85 #93-1
A 4=#93 #99-1
A 5=#99 #105-1
A 6=#105 e000:ffff
A f=f000:0 f000:ffff
```

Now execute this macro to define the address ranges.

```
-EM MEMTESTADDR
```

These ranges measure program activity in the procedure which does the memory test. Note the use of the #linenumber-1 format to set the end of the range to the end of the previous linenumber. This program has a wait loop which will fill up the PA measurement samples. To eliminate wait loop samples during the measurement, exclude ranges are put in the PA measurement example below. Now the sampling can be started at the current CS:IP. The terminate condition is set to line #105. During the measurement, the program will ask you for a start address and length for the memory test program. Fill in the answers as shown in the following example.

```
-PA
PA-GO FROM-<enter>
PA-STOP AT-#105
PA-STOP AT-<enter>
PA-EXCLUDE-0:0 .initialize
PA-EXCLUDE-#105 e000:ffff
PA-SAMPLE RATE-<enter>
SAMPLE RATE IS:  0.100 ms
STRIKE ANY KEY TO BEGIN SAMPLING.
SAMPLING BEGUN.

WHAT K WOULD YOU LIKE TO START TESTING (0 -1023)? 200
HOW MANY K WOULD YOU LIKE TO TEST (1-960)? 5

Sampling terminated at 09BF:01CF=..MEMORY_TESTER#105
        Program Activity Measurement
        Sample rate is:  0.100 ms

Min address         Max address          Count    %   0   20   40   60   80   100
0000:0000           .INITIALIZE          excluded from sampling
..MEMORY_TE#105     E000:FFFF            excluded from sampling
0000:0000           .INITIALIZE          0        0   |
..MEMORY_TES#75     ..MEMOR#76+000B      2        <1  |
..MEMORY_TES#80     ..MEMOR#83+000B      921      48  |***********
..MEMORY_TES#85     ..MEMOR#89+005B      201      10  |**
..MEMORY_TES#93     ..MEMOR#94+0002      215      11  |**
..MEMORY_TES#99     ..MEMO#103+0002      1        <1  |
..MEMORY_TE#105     E000:FFFF            0        0   |
F000:0000           F000:FFFF            555      29  |*****
                                         Total    98
```

NOTE: The data in this histogram may vary according to the specific attributes of your system.

It appears that half of the time in this procedure is spent between linenumbers 80 and 85; therefore, if it desired to make this program faster, this would be the area to work on.

Next measure the time to test each one K block of memory. The Program Timing Analyzer command will be used to get a sample of these times. Set the timed events to be at the top and the bottom of the loop. First re-initialize the CS:IP.

```
-INI
-PTA
PTA-GO FROM-<enter>
PTA-STOP AT-#105
PTA-STOP AT-
PTA-EVENT ADDRESS-#76
PTA-EVENT ADDRESS-#103
PTA-EVENT ADDRESS-<enter>
STRIKE ANY KEY TO BEGIN SAMPLING.
SAMPLING BEGUN.

WHAT K WOULD YOU LIKE TO START TESTING (0-1023)? 200
HOW MANY K WOULD YOU LIKE TO TEST (1-960)? 5

Sampling terminated at 09BF:01CF=..MEMORY_TESTER#105
        Program Timing Analysis
        Time base is:  RELATIVE
        Last sample is: 12

Sample# Address                    Time
1       ..MEMORY_TESTER#76          3.587 sec
2       ..MEMORY_TESTER#103        24.600 ms
3       ..MEMORY_TESTER#76          0.000 ms
4       ..MEMORY_TESTER#103        24.500 ms
5       ..MEMORY_TESTER#76          0.000 ms
6       ..MEMORY_TESTER#103        24.500 ms
7       ..MEMORY_TESTER#76          0.000 ms
8       ..MEMORY_TESTER#103        24.500 ms
9       ..MEMORY_TESTER#76          0.000 ms
10      ..MEMORY_TESTER#103        24.600 ms
11      ..MEMORY_TESTER#76          0.000 ms
12      ..MEMORY_TESTER#105         0.000 ms
        PTA- E
```

NOTE: The data in this histogram may vary according to the specific attributes of your system.

The loop executes between 24.5 and 24.6 ms. Note that to get out of this display, you type E.

Next, the Procedure Duration measurement will be used to make a similar measurement. Use the "US" macro to set the time ranges in 100 microsecond increments starting at 24 milliseconds. Note that since the increments are being set at the maximum resolution of the timer (0.1 ms), that the minimum and maximum times for each range are set the same when the PD measurement histogram displays.

```
-EM US 24
-T 0=024.0 024.0
-T 1=024.1 024.1
-T 2=024.2 024.2
-T 3=024.3 024.3
-T 4=024.4 024.4
-T 5=024.5 024.5
-T 6=024.6 024.6
-T 7=024.7 024.7
-EM us2 24
-T 8=024.8 024.8
-T 9=024.9 024.9
-T a=1+024.0 1+024.0
-T b=1+024.1 1+024.1
-T c=1+024.2 1+024.2
-T d=1+024.3 1+024.3
-T e=1+024.4 1+024.4
-T f=1+024.5 i
```

Now do a Procedure Duration measurement to determine the range of execution times for this procedure. Restart the program first.

```
-INI
-PD
PD-GO FROM-<enter>
PD-STOP AT-#105
PD-STOP AT-<enter>
PD-TIMER ON AT-#76
PD-TIMER ON AT-<enter>
PD-TIMER OFF AT-#103
PD-TIMER OFF AT-<enter>
STRIKE ANY KEY TO BEGIN SAMPLING.
SAMPLING BEGUN.

WHAT K WOULD YOU LIKE TO START TESTING (0-1023)? 200
HOW MANY K WOULD YOU LIKE TO TEST (1-960)? 5

Sampling terminated at 09BF:01CF=..MEMORY_TESTER#105
      Procedure Duration Measurement
```

```
Min Time        Max Time        Count   %    0   20  40  60  80  100
24.000 ms       24.000 ms       0       0    |
24.100 ms       24.100 ms       0       0    |
24.200 ms       24.200 ms       0       0    |
24.300 ms       24.300 ms       0       0    |
24.400 ms       24.400 ms       0       0    |
24.500 ms       24.500 ms       3       60   |******************
24.600 ms       24.600 ms       2       40   |*************
24.700 ms       24.700 ms       0       0    |
24.800 ms       24.800 ms       0       0    |
24.900 ms       24.900 ms       0       0    |
25.000 ms       25.000 ms       0       0    |
25.100 ms       25.100 ms       0       0    |
25.200 ms       25.200 ms       0       0    |
25.300 ms       25.300 ms       0       0    |
25.400 ms       25.400 ms       0       0    |
25.500 ms       Infinity        0       0    |
                                Total   100
```

NOTE: The data in this histogram may vary according to the specific attributes of your system.

Since this is a very tight loop, all samples fall within a very narrow range as was expected from the PTA.

Now use the event counter mode to count the occurrences of the program events. The events in this measurement will be the number of times through the loop at various points.

```
-INI
-PE
PEC-GO FROM-<enter>
PEC-STOP AT-#105
PEC-STOP AT-<enter>
PEC-EVENT ADDRESS-#76, #80, #89
PEC-EVENT ADDRESS-#93, #102
PEC-EVENT ADDRESS-<enter>
STRIKE ANY KEY TO BEGIN SAMPLING.
SAMPLING BEGUN.

WHAT K WOULD YOU LIKE TO START TESTING (0-1023)? 200
HOW MANY K WOULD YOU LIKE TO TEST (1-960)? 5

Sampling terminated at 09BF:01CF=..MEMORY_TESTER#105
          Program Event Counter

Address                                  Count   %   0   20   40   60   80   100
..MEMORY_TESTER#105                      1       <1  |
..MEMORY_TESTER#76                       6       <1  |
..MEMORY_TESTER#80                       5125    50  |***************
..MEMORY_TESTER#89                       0       0   |
..MEMORY_TESTER#93                       5120    49  |***************
..MEMORY_TESTER#102                      5       <1  |
```

NOTE: The data in this histogram may vary according to the specific attributes of your system.

The loops were supposed to be entered 1024 times for each block of memory tested. Since 5 blocks were tested, use the EValuate command to verify the counts achieved in the PE counter.

```
          -EV 1024t * 5
          1400H 5120T +5120T 1010000000000Y '..'
```

These numbers look correct since line #93 was executed exactly 5120 times.

CHAPTER 4
SPTA COMMAND REFERENCE

INTRODUCTION

The SPTA commands are summarized in the following list. Many of the SPTA commands are identical to the PROBE commands; therefore, the descriptions and examples for these commands are not repeated in this chapter, but can be found in Chapter 6 of the PROBE manual. In the following list, the SPTA commands that are related to performance measurement are indicated with a single asterisk, and these commands are described in this chapter. The commands with two asterisks are commands for the stand-alone version of the SPTA.

COMMAND	**DESCRIPTION**
Arange*	Assign address ranges for PAM command
Asm**	Assemble instructions into memory
BP**	Define or display sticky breakpoints
BYte	Display/change bytes in memory
COMpare	Compare blocks of memory
CONsole	Assign local/remote console
DElete	Delete symbols or macros
ECho	Enable/disable display of macro execution
EMacro	Execute macro and pass it parameters
EVal	Evaluate expression
FIll	Fill memory with a string
FLag	Display/change flags in memory
Go	Go and set breakpoints
IF	If/then/else construct
INIt	Set registers to program load conditions
INTerrupt	Enable/disable selected interrupt lines
LIst	List output to disk file or line printer
LOAd	Load program or symbol table
LOOp	Loop count construct
MAcro	Define/display macros
MEnu	Enable/disable menu
MODule	Assign default module or display modules
MORe	Display next menu
MOVe	Move a block of memory

* Command description is found in this chapter, all other command descriptions are found in the PROBE manual.

** Command found only in the stand-alone version of SPTA. The command description is found in the PROBE manual.

COMMAND	**DESCRIPTION**
NEst	Display procedure calling sequence
NOVerif	Suppress read after write of byte, word, ptr
PAm*	Do program activity measurement
PDm*	Do procedure duration measurement
PEc*	Do program event count measurement
POrt	Display/change IO ports
PRint	Print expressions
PTA*	Do procedure timing analysis
PTr	Display/change pointers in memory
Quit	Quit PERF and return to DOS
Register	Display/change registers
SAve	Save memory or macros
SCreen	Enable screen switching
SEArch	Search memory for a string
SELect	Select modules for symbol table
STep	Single step program execution
SYmbol	Display or define symbols
Trange*	Define/display time ranges for PDM measurement
Unassemble	Unassemble instructions
WIndow	Open window for display of information
WOrd	Display/change words in memory

* Command description is found in this chapter, all other command descriptions are found in the PROBE manual.

** Command found only in the stand-alone version of SPTA. The command description is found in the PROBE manual.

ARANGE

PURPOSE: To define an address range and assign it a number for use during the histogram analysis of a PA measurement. The command also displays the range assigned to rangenumbers.

FORMAT: **A rangenumber =<address 1> <address 2>**
=<address 1> Length <length>
=<enter> to delete

REMARKS: **Rangenumber** is a number from 0 thru F. An address range of either of the first two forms shown above is assigned to each rangenumber. An address range can be deleted from a rangenumber by the third form shown above. This has the effect of eliminating this rangenumber from the PA measurement histogram display. The rangenumbers are initially assigned to a unique 64k block of memory.

FORMAT: **A [rangenumber]**

REMARKS: The range assigned to the specified **rangenumber** is displayed. If no **rangenumber** is specified, then the definitions for all rangenumbers are displayed.

EXAMPLES: The following command displays all range numbers. The the default ranges are shown.

```
-A
Range 0.   0000:0000   0000:FFFF
Range 1.   1000:0000   1000:FFFF
Range 2.   2000:0000   2000:FFFF
Range 3.   3000:0000   3000:FFFF
Range 4.   4000:0000   4000:FFFF
Range 5.   5000:0000   5000:FFFF
Range 6.   6000:0000   6000:FFFF
Range 7.   7000:0000   7000:FFFF
Range 8.   8000:0000   8000:FFFF
Range 9.   9000:0000   9000:FFFF
Range A.   A000:0000   A000:FFFF
Range B.   B000:0000   B000:FFFF
Range C.   C000:0000   C000:FFFF
Range D.   D000:0000   D000:FFFF
Range E.   E000:0000   E000:FFFF
Range F.   F000:0000   F000:FFFF
```

To assign ranges for a PA measurement using the symbols of the program to be measured:

```
A 0=0:0 .INITIALIZE
A 1=#75 #80-1
A 2=#80 #85-1
A 3=#85 #93-1
A 4=#93 #99-1
A 5=#99 #105-1
A F=F000:0 F000:FFFF
```

To delete the range for rangenumber 6 from the histogram display:

```
A 6=<enter>
```

TRANGE

PURPOSE: To define a time range and assign it a number for use during the histogram display of a PD measurement. The command also displays the range assigned to rangenumbers.

FORMAT:

```
-T rangenumber = <time 1> <time 2>
               = <time 1> I  for infinity
               = <CR> to delete
```

REMARKS: **Rangenumber** is a number from 0 thru F. A time range of either of the first two forms shown above is assigned to each rangenumber. A time range can be deleted from a rangenumber by the third form shown above. This has the effect of eliminating this rangenumber from the PD measurement histogram display.

The timer resolution is 0.1 millisecond. Time can be specified as follows:

FORMAT	TIMEBASE
xxx.x or xxx.x m	-milliseconds
xx.xxx s	-seconds
xx:xx	-minutes:seconds
xx:xx:xx	-hours:minutes:seconds
I	-infinity (119 hrs)

FORMAT: **T [rangenumber]**

REMARKS: The range assigned to the specified **rangenumber** is displayed. If no **rangenumber** is specified, then the definitions for all rangenumbers are displayed.

EXAMPLES: The following command displays all range numbers. The default ranges are shown.

```
-T
Range 0.   0.000 ms     0.900 ms
Range 1.   1.000 ms     1.900 ms
Range 2.   2.000 ms     2.900 ms
Range 3.   3.000 ms     3.900 ms
Range 4.   4.000 ms     4.900 ms
Range 5.   5.000 ms     5.900 ms
Range 6.   6.000 ms     6.900 ms
Range 7.   7.000 ms     7.900 ms
Range 8.   8.000 ms     8.900 ms
Range 9.   9.000 ms     9.900 ms
Range A.   10.000 ms    10.900 ms
Range B.   11.000 ms    11.900 ms
Range C.   12.000 ms    12.900 ms
Range D.   13.000 ms    13.900 ms
Range E.   14.000 ms    14.900 ms
Range F.   15.000 ms    Infinity
```

To define a group of time ranges:

```
COMMAND                 NOTES
-T 0=0.3m 0.5m          ;300 to 500 microseconds
-T 1=100.9 200.0        ;100.9 to 200.0 milliseconds
-T 2=10.9s 20.0s        ;10.9 to 20.0 seconds
-T 3=37.963s 44.332s    ;37.963 to 44.332 seconds
-T 4=3:4 10:10          ;3 minutes and 4 seconds to
                        ;10 minutes and 10 seconds
-T 5=55:20 55:21        ;55 minutes and 20 seconds to
                        ;55 minutes and 21 seconds
-T 6=3:20:5 4:0:0       ;3 hours, 20 minutes and
                        ;5 seconds to 4 hours
```

PROGRAM ACTIVITY MEASUREMENT

PURPOSE: To measure and display where a program spends its time.

FORMAT:

PA
PA- GO FROM -
PA- STOP AT -
PA- STOP AT -
PA- EXCLUDE -
PA- EXCLUDE -
PA- SAMPLE RATE-

REMARKS: The Program Activity measurement samples the CS:IP during instruction execution and displays a histogram of the % execution time spent in specified address ranges over all samples taken. The Program Activity Measurement prompts you for the following parameters.

PA- GO FROM -
This prompts for the start address of program execution. It defaults to the current CS:IP if no address is specified.

PA- STOP AT -
This prompts for the measurement termination condition. It may be desirable to have several terminate addresses; therefore, the STOP AT prompt reiterates until no address is supplied. The measurement will also terminate when the maximum number of samples is collected (see Appendix C), the STOP button is pressed, or the program terminates.

PA- EXCLUDE -
This prompts for a range of memory which will be excluded from the PA measurement. Samples which are taken in this area are ignored. Two exclude regions are available.

PA- SAMPLE RATE -
Samples are multiples of 100 microsecond times. If no rate is supplied, then the default is 100 microseconds.

FORMAT: **PA D**

REMARKS: This command displays a histogram of the samples by the PA measurement as a % of execution time for each range over all collected samples and is automatically executed after each PA measurement. This command allows the ranges to be changed with the Arange command for a different histogram display for a given PA measurement data collection.

EXAMPLES: This PA command starts at the current CS:IP and stops at linenumber 105. The default sample rate is used.

```
-PA
PA- GO FROM - <enter>
PA- STOP AT - #105
PA- STOP AT - <enter>
PA- EXCLUDE - <enter>
PA- EXCLUDE - <enter>
PA- SAMPLE RATE- <enter>
SAMPLE RATE IS: 0.100 MS
STRIKE ANY KEY TO BEGIN SAMPLING. <enter>
SAMPLING BEGUN
SAMPLING TERMINATED, CS:IP= ..MEMORY_TESTER#105
        Program Activity Measurement
        Sample rate is:  0.100 ms

Min address         Max address          Count    %   0   20  40  60  80  100
0000:0000           .INITIALIZE          excluded from sampling
..MEMORY_TE#105     E000:FFFF            excluded from sampling
0000:0000           .INITIALIZE          0        0   |
..MEMORY_TES#75     ..MEMOR#76+000B      2        <1  |
..MEMORY_TES#80     ..MEMOR#83+000B      921      48  |***********
..MEMORY_TES#85     ..MEMOR#89+005B      201      10  |**
..MEMORY_TES#93     ..MEMOR#94+0002      215      11  |**
..MEMORY_TES#99     ..MEMO#103+0002      1        <1  |
..MEMORY_TE#105     E000:FFFF            0        0   |
F000:0000           F000:FFFF            555      29  |*****
                                         Total:   98
```

PROCEDURE DURATION MEASUREMENT

PURPOSE: To measure the time between the execution of specified instructions.

FORMAT:

PD
PD- GO FROM -
PD- STOP AT -
PD- STOP AT -
PD- TIMER ON AT -
PD- TIMER ON AT -
PD- TIMER OFF AT -
PD- TIMER OFF AT -

REMARKS: The Procedure Duration measurement measures the time between turning a timer on and then off at specified locations. The measurement is accomplished by setting breakpoints at each timer on/timer off specification. A sample is defined as the time to execute from the timer on breakpoint to the timer off breakpoint. The measurement generates a histogram of the collected time samples in the specified time ranges set with the Trange command over all time samples collected. Therefore, if the sum total % of all ranges in the displayed histogram does not add up to 100%, then there are other time samples collected which are not displayed. Time distribution measurements allow characterization and verification of best- and worst-case execution times. By highlighting modules consuming inordinate amounts of processing time, sources of overall system degradation are identified. Spurious execution times, as a result of faulty module execution or other outside module influences, become highly visible in this measurement. This measurement prompts you for the following parameters.

PD- GO FROM -
Program execution starts here. If the start address is not specified, the current CS:IP is assumed.

PD- STOP AT -
This prompts for the termination condition for the measurement. It may be desirable to have several terminate addresses; therefore, the STOP AT prompt reiterates until no address is supplied. The measurement will also terminate when the maximum number of samples is collected (see Appendix C), the STOP button is pressed, or the end of the program is reached.

PD- TIMER ON AT -
This prompts for the address where the timer is to be started. It may be desirable to start the timer at a number of locations; therefore, this prompt reiterates until no address is supplied.

PD- TIMER OFF AT -
This prompts for the address where the timer is to be stopped. It may be desirable to stop the timer at a number of locations; therefore, this prompt reiterates until no address is supplied.

FORMAT: **PD D**

REMARKS: The histogram display for the PD command is generated automatically after each PD measurement, but it can also be generated by this command. Initial Procedure Duration measurements often span a relatively large total time interval. As the investigation proceeds, it is possible to focus on specific time events, by using smaller time ranges defined more closely around pertinent time values using the Trange command. The resulting magnification of data provides better measurement resolution.

EXAMPLE: This example starts program execution at the current CS:IP and measures the time between line numbers 76 and 103.

```
-PD
PD- GO FROM - <enter>
PD- STOP AT - #105
PD- STOP AT - <enter>
PD- TIMER ON AT - #76
PD- TIMER ON AT - <enter>
PD- TIMER OFF AT - #103
PD- TIMER OFF AT - <enter>
STRIKE ANY KEY TO BEGIN SAMPLING. <enter>
SAMPLING BEGUN
SAMPLING TERMINATED, CS:IP=..MEMORY_TESTER#105
      Procedure Duration Measurement

Min Time        Max Time        Count   %    0   20  40  60  80  100
24.000 ms       24.000 ms       0       0    |
24.100 ms       24.100 ms       0       0    |
24.200 ms       24.200 ms       0       0    |
24.300 ms       24.300 ms       0       0    |
24.400 ms       24.400 ms       0       0    |
24.500 ms       24.500 ms       3       60   |******************
24.600 ms       24.600 ms       2       40   |*************
24.700 ms       24.700 ms       0       0    |
24.800 ms       24.800 ms       0       0    |
24.900 ms       24.900 ms       0       0    |
25.000 ms       25.000 ms       0       0    |
25.100 ms       25.100 ms       0       0    |
25.200 ms       25.200 ms       0       0    |
25.300 ms       25.300 ms       0       0    |
25.400 ms       25.400 ms       0       0    |
25.500 ms       Infinity        0       0    |
                                Total:  100
```

PROGRAM EVENT COUNTER

PURPOSE: To count the number of times a specified instruction is executed and display a histogram of these events.

FORMAT: **PE**
PE- GO FROM -
PE- STOP AT -
PE- STOP AT -
PE- EVENT ADDRESS -
PE- EVENT ADDRESS -

REMARKS: The Program Event counter measures the number of times the specified event occurs and displays a histogram of the % of occurrence of each event over all events. This measurement prompts you for the following parameters.

PE- GO FROM -
Program execution starts here. If no address is specified, then the current CS:IP is assumed.

PE- STOP AT -
This prompts for the termination condition for the measurement. It may be desirable to have several terminate addresses. Therefore, the STOP AT prompt reiterates until no address is supplied. The measurement will also terminate when the maximum number of samples is collected (see Appendix C), the STOP button is pressed, or the end of the program is reached.

PE- EVENT ADDRESS -
Up to 16 events may be specified. The prompt reiterates until no address is supplied.

FORMAT: **PE D**

REMARKS: This command displays the PE data collected with it sorted in ascending order based on the address. This command is automatically executed at the end of a PE measurement.

EXAMPLE: This command starts execution at the current CS:IP and specifies 5 line numbers for events.

```
-PE
PE- GO FROM - <enter>
PE- STOP AT - #105
PE- STOP AT - <enter>
PE- EVENT ADDRESS - #76
PE- EVENT ADDRESS - #80
PE- EVENT ADDRESS - #89
PE- EVENT ADDRESS - #91
PE- EVENT ADDRESS - #102
PE- EVENT ADDRESS - <enter>
STRIKE ANY KEY TO BEGIN SAMPLING.  <enter>
SAMPLING BEGUN

PROGRAM EVENT COUNTER
TERMINATED BY:  EXECUTED ..MEMORY_TESTER#105
      Program Event Counter

Address                            Count    %    0   20   40   60   80   100
..MEMORY_TESTER#105                1        <1   |
..MEMORY_TESTER#76                 6        <1   |
..MEMORY_TESTER#80                 5125     50   |***************
..MEMORY_TESTER#89                 0        0    |
..MEMORY_TESTER#93                 5120     49   |***************
..MEMORY_TESTER#102                5        <1   |
```

PROCEDURE TIMING ANALYSIS

PURPOSE: To perform a time measurement on a single event or a string of events.

FORMAT: **PTA**
PTA- GO FROM -
PTA- STOP AT -
PTA- STOP AT -
PTA- EVENT ADDRESS -
PTA- EVENT ADDRESS -

REMARKS: This measurement shows a linear timed sequence of program execution and can provide a starting point for choosing the time ranges in the histograms. This command measures the time between defined events. The events are defined as executed instructions, and a maximum of 16 events can be specified at one time.

The timer is read when each event occurs. In the display, the relative time between events or the absolute time of each event is shown. The time shown for the first event is always 0. Relative time is the time from the previous event to the current event. Absolute time is the time from the first event to the current event. By specifying only one event, repeat occurrences of the same event is measured. This will allow problems such as interrupt saturation to be found by showing occurrences of an event which may delay other events.

This measurement prompts you for the following parameters:

PTA- GO FROM -
This is the start address for program execution. If no start address is given the current CS:IP is assumed.

PTA- STOP AT -
This prompts for the termination condition for the measurement. It may be desirable to have several terminate addresses; therefore, the STOP AT prompt reiterates until no address is supplied. The measurement will also terminate when the maximum number of samples is collected (see Appendix C), the STOP button is pressed, or the end of the program is reached.

PTA- EVENT ADDRESS -
Up to 16 events may be specified. The prompt reiterates until no address is supplied.

At the end of the measurement, the first samples are displayed. The following control keys are shown in a PTA menu window so they will not have to be remembered. You may specify the display to start at a new sample by typing "S" followed by a number in response to this prompt. If <PgDn> is typed, then the next page of samples is displayed. Ctrl <PgUp> displays the first page of samples. Ctrl <PgDn> displays the last page of samples. If "R" is typed, the current page will be re-displayed in relative times. If "A" is typed, the current page will be re-displayed in absolute times. E key will terminate the display of additional samples.

FORMAT: **PTA D**

REMARKS: The data collected during the PTA measurement can be displayed. The display is shown automatically at the end of the measurement. The display formats are Relative or Absolute.

EXAMPLE: This example PTA command starts execution at the current CS:IP and defines two events.

```
PTA <enter>
PTA- GO FROM - <enter>
PTA- STOP AT - #105
PTA- STOP AT - <enter>
PTA- EVENT ADDRESS - #76
PTA- EVENT ADDRESS - #103
PTA- EVENT ADDRESS - <enter>
STRIKE ANY KEY TO BEGIN SAMPLING.  <enter>
SAMPLING BEGUN.

SAMPLING TERMINATED, CS:IP= ..MEMORY_TESTER#105
        Program Timing Analysis
        Time base is:  RELATIVE
        Last sample is: 12

Sample#     Address                    Time
1           ..MEMORY_TESTER#76        3.587 sec
2           ..MEMORY_TESTER#103      24.600 ms
3           ..MEMORY_TESTER#76        0.000 ms
4           ..MEMORY_TESTER#103      24.500 ms
5           ..MEMORY_TESTER#76        0.000 ms
6           ..MEMORY_TESTER#103      24.500 ms
7           ..MEMORY_TESTER#76        0.000 ms
8           ..MEMORY_TESTER#103      24.500 ms
9           ..MEMORY_TESTER#76        0.000 ms
10          ..MEMORY_TESTER#103      24.600 ms
11          ..MEMORY_TESTER#76        0.000 ms
12          ..MEMORY_TESTER#105       0.000 ms
            PTA- "E"
```

EXAMPLE: The data can also be displayed in absolute time mode with the following command.

```
PTA -A
PROGRAM TIMING ANALYSIS
TIME BASE IS :  RELATIVE
LAST SAMPLE IS:  5

Sample#     Address               Time
1           ..MEMORY_TE#76         0.000 MS
2           ..MEMORY_T#103         3.587 MS
3           ..MEMORY_TE#76        28.187 MS
4           ..MEMORY_T#103        28.187 MS
5           ..MEMORY_TE#76        52.687 MS
6           ..MEMORY_T#103        52.687 MS
7           ..MEMORY_T#76         77.187
            PTA-
```

APPENDICES

APPENDIX TITLE

APPENDIX A
SPTA ERROR MESSAGES

ARANGE

"Must be: 0<= rangenum <=F"
Rangenumber is 0 to F.

PA Measurement

"Must be: 0.100 <= sample rate <= 00.27:27"
This is the allowable range for the sample rate.

TRANGE

"Must be: 0 <= rangenum <= F"
Rangenumber is 0 to F.

PA Measurement, PD Measurement, PE Counter, and PTA

"Samples not collected in this mode"
For example, the PA D command cannot display the samples if they were collected in a PTA measurement.

APPENDIX B
FILES ON YOUR SPTA DISKETTE

There are several files on your SPTA diskette which may or may not be needed depending upon what you are doing. A list of these files and their purpose is given below.

FILE NAME	DESCRIPTION	USED FOR
PERF.EXE	SPTA SOFTWARE	RUNNING SPTA
PAMNHW.EXE	SPTA SOFTWARE	RUNNING SPTA
PROBE.CFG	CONFIG FILE	BASE ADDRESS, CONSOLE
PERF.MAC	USEFUL MACROS	OPTIONAL MACROS
MEMTEST.PAS	DEMO SOURCE	SPTA DEMO
MEMTEST.EXE	DEMO EXECUTABLE	SPTA DEMO
MEMTEST.MP1	DEMO MAP FILE	SPTA DEMO

If you have the stand-alone version of the Software Performance and Timing Analyzer, then your diskette contains the files listed below.

FILE NAME	DESCRIPTION	USED FOR
SWPERF.EXE	SW PERF SOFTWARE	RUNNING SW PERF
PERF.MAC	USEFUL SPTA MACROS	OPTIONAL MACROS
PROBE.CFG	CONFIG FILE	CONSOLE
STRIP.EXE	REMOVE MAP SYMBOLS	REMOVE MAP SYMBOLS
STRIPP.EXE	REMOVE MAP SYMBOLS	XLATE PLINK MAP TO DOS
STRIPPE.EXE	REMOVE MAP SYMBOLS	XLATE PLINK EXE TO DOS
STRIP.C	C LANGUAGE DATA FILE	STRIPPING SYMBOLS
STRIP.MSC	MICROSOFT C DATA FILE	STRIPPING SYMBOLS
STRIP10.PAS	MICROSOFT PASCAL DATA FILE	STRIPPING SYMBOLS
VT100SW.CFG	VT100 CONFIG FILE	VT100 CONSOLE
ADMSASW.CFG	ADM3A CONFIG FILE	ADM3A CONSOLE
TV925SW.CFG	TELEVIDEO 925 CONFIG FILE	TELEVIDEO 925 CONSOLE
DUMBSW.CFG	DUMB TERM. CONFIG FILE	DUMB TERMINAL
PC.CFG	PC CONFIG FILE	PC USED AS TERM.
TERMCOM.EXE	TERMINAL SOFTWARE	MAKES PC INTO TERMINAL
CLIST.EXE	LISTING PRODUCER	CREATE C LANG. LISTING

A backup of the SPTA software should be made first. In a floppy based system, it may not be possible to have all of the above files in addition to the user files on one floppy. Using the descriptions above, make a working copy of the SPTA software by copying only those files required to run the SPTA.

APPENDIX C
TECHNICAL REPORTS

This section provides information on potential problem areas, technical information, and bug reports for the Software Performance and Timing Analyzer (SPTA). This section will be updated periodically by ATRON for customers who have sent their registration cards in and who are in their first year warranty period. The reports will also be updated for those who have purchased extended warranty support.

This appendix contains the following technical reports.

SPTA MEASUREMENT SAMPLES
INTERPRETATION OF REPORTED TIMES
PROBE TIMER
MODULE USAGE MEASUREMENT
INTERMODULE DURATION MEASUREMENT

Last revised: 5/2/86

SPTA MEASUREMENT SAMPLES

The samples taken by the SPTA occur at breakpoints set by the measurement being taken and are intrusive into the total execution time of the applications program. The overhead per sample is shown in the table below. Also shown is the maximum number of samples allowable for each measurement.

Table C-1. SPTA Measurement Samples

Measurement type	Intrusion per sample	Maximum # samples
PA	4.5 ms	8188
PD	11.9 ms	8188
PE	4.5 ms	65535
PTA	6.6 ms	4094

INTERPRETATION OF REPORTED TIMES

The maximum resolution of the timer is 0.1 ms. Therefore, events which occur between 0.1 ms increments are reported as follows:

Reported Time	Interpretation
0.1 ms	execution took from 0 to 100 us
0.2 ms	execution took from 101 to 200 us
0.3 ms	execution took from 201 to 300 us
.	.
.	.
.	.
100.5ms	execution took from 100,401 to 100,500 us
.	.
.	.
.	.

PROBE TIMER

The timer used for the PC PROBE based versions of SPTA is located on the PC PROBE board and is not accessable by the user. The timer used for the stand-alone version of SPTA is Timer #0 on the mother board and is addressed at ports 40 and 43. Critical regions will not be seen.

MODULE USAGE MEASUREMENT

The module usage measurement is an application of the Procedure Duration measurement mode, and it provides a distribution of the time available for execution of other tasks after a specific module executes. This measurement indicates the intensity of demand for the services of a module. This results in an extremely useful measurement for identifying program areas where optimization efforts can be most effective. The module usage measurement is the complement of the procedure duration measurement in that it measures the time from module completion to the time that same module is used again. Figure C-1 contrasts the two measurements.

```
Procedure                               Module
Duration                                Usage
Measurement                             Measurement
                     |                  |
                     |                  |
                     |                  |
timer starts--->     |                  |<--- timer stops
                     |                  |
                     |  module          |
                     |  tested          |
                     |                  |
timer stops--->      |                  |<---timer starts
                     |                  |
                     |                  |
                     |                  |
```

Figure C-1. Procedure Duration and Module Usage Measurements

In a typical application, the Module Usage measurement may reflect low demand for a specific module, allowing other scheduled tasks to occur normally. A small percentage of the time, the measurement may indicate heavy module usage, preventing other system tasks from being performed at all. This usage measurement is a valuable pointer to task scheduling problems, indicating the need for operating system interface optimization. Program modifications can then produce more effective and efficient task scheduling, resulting in greater overall system throughput.

INTERMODULE DURATION MEASUREMENT

The intermodule duration measurement is an application of the Procedure Duration measurement and produces a distribution of the time intervals between successive execution of two specified modules. The time duration is measured between the execution of the last instruction of the "from" module and the first execution of the "to" module. This is illustrated in Figure C-2.

```
|              |--->Module A <--timer starts
| main         |<---        |
| module       | intermodule duration
|              |            |
|              |--->Module B <--timer stops
|              |<---
|              |
```

Figure C-2. Intermodule Duration Measurement

This time measurement provides a basis for resolving module interaction problems. Transfer time between modules is often critical. Overall system performance can be greatly improved by first identifying worst-case program paths, and then optimizing the interaction involved.

INDEX