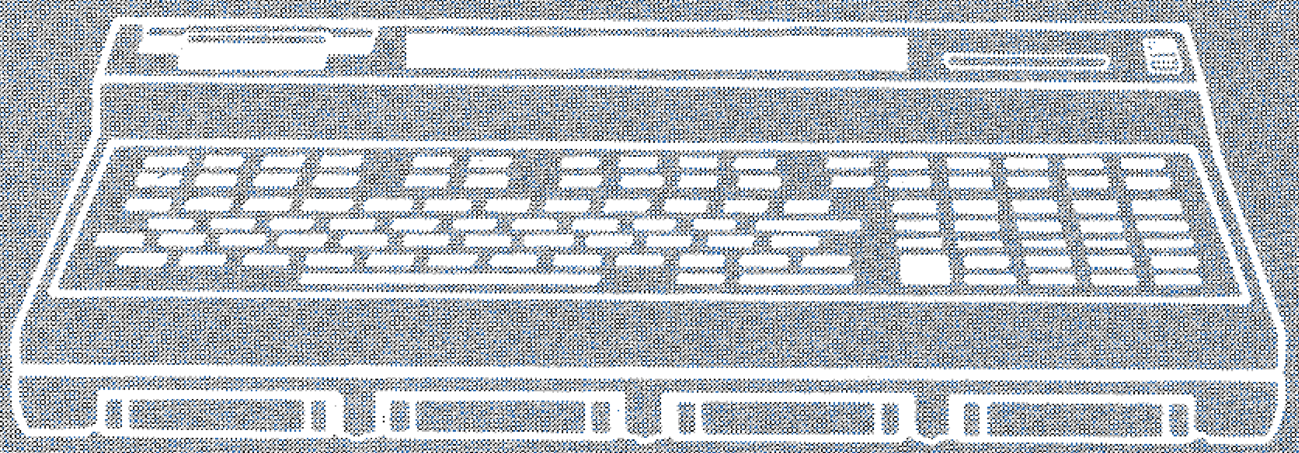
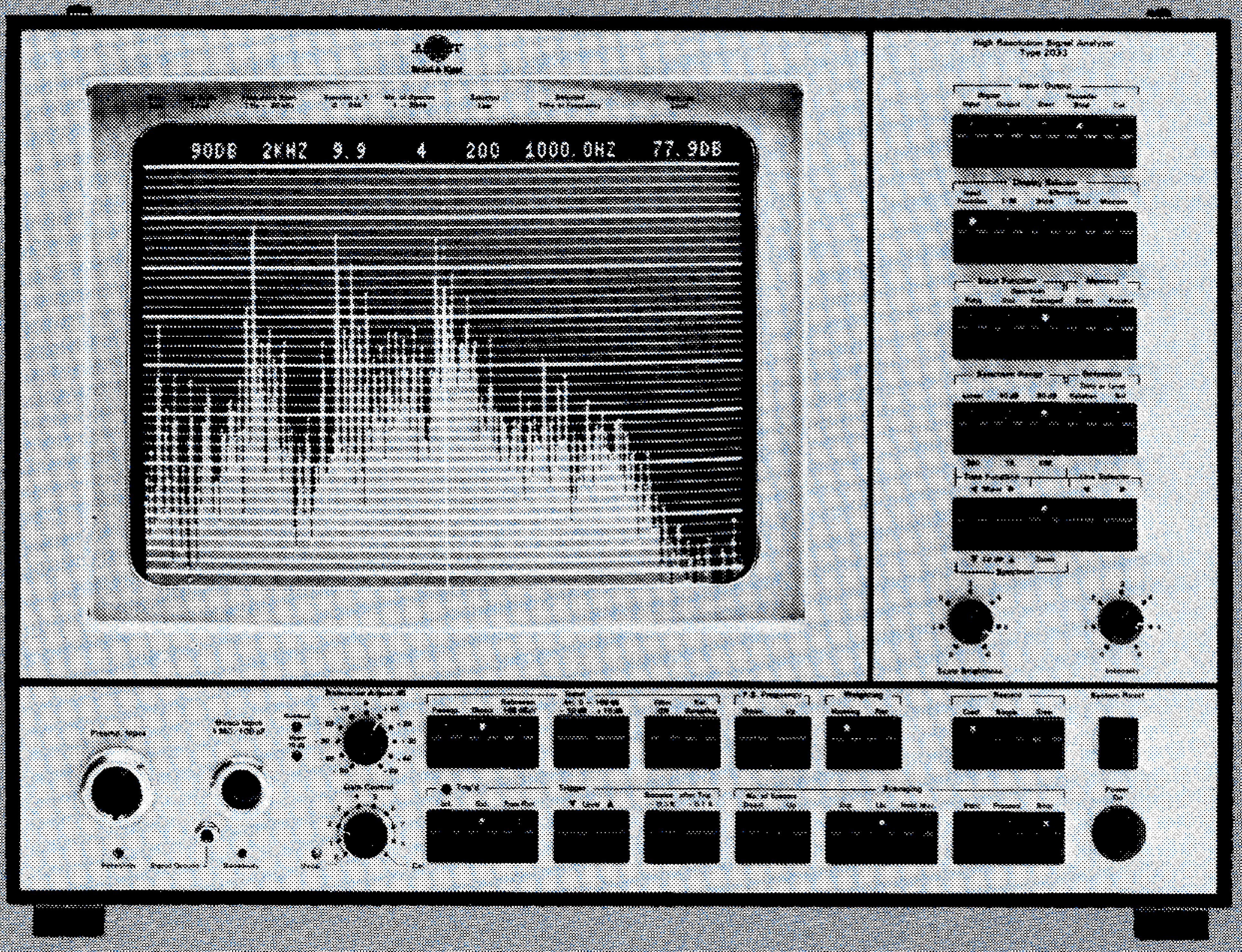
BRUEL & KJAER application notes

Use of the High Resolution Signal Analyzer Type 2033 with a Desk-top Calculator

```
0: dim A$[20+1+16],B$[20],A[20]
1: buf "IN",A$,3
2: 1000→M
3: 10→L
4: wtb 725,"#3,16382,1;"
5: fmt 1,z,b,b,b
6: red 725.1,A,B,C
7: 256*A+B→A
8: (A+M-1-4096)mod10240+4096→S
9: (A+M+L-2-4096)mod10240+4096→T
10: if S>T;gto "CASE-II"
11: "CASE-I":
12: fxd 0;wtb 725,"#3,",str(S),","
13: gsb "DUMP-7"
14: A$[1,2L]→B$[1,2L]
15: gto "CONV"
16: "CASE-II":
17: 14336-S→U
18: L-U→V
19: fxd 0;wtb 725,"#3,",str(S),","
20: gsb "DUMP-7"
21: A$[1,2U]→B$[1,2U]
22: fxd 0;wtb 725,"#3,4096,",str(V)
23: gsb "DUMP-7"
24: A$[1,2V]→B$[2U+1,2L]
25: "CONV":
26: gsb "CAL-7"
27: gsb "CONVERT-7"
28: end
29:
30: "DUMP-7":
31: buf "IN"
32: tfr 725,"IN"
33: jmp rds("IN")#-1
34: ret
35:
36: "CAL-7":
37: wtb 725,"#4,D;"
38: red 725,D
39: wtb 725,"#4,√;"
40: red 725,E
41: 10*((10*(D+E)+19)/20)→F
42: ret
43:
44: "CONVERT-7":
45: F/32768*1e-6→K
46: for I=1 to L
47: K*itf(B$[2I-1,2I])→A[I]
48: next I
49: ret
*5262
```





089-81

USE OF THE HIGH RESOLUTION SIGNAL ANALYZER TYPE 2033 WITH A DESK-TOP CALCULATOR

Programming examples

by *Jens Hee*, Brüel & Kjær

INTRODUCTION

In applications where automatic spectral analysis is desirable remote programmability of the spectrum analyzer is necessary.

The IEC/IEEE interface of the 2033 gives the possibility of complete remote control of the analyzer and I/O from the alphanumeric text line and any part of the main memory. As the data format used in the main memory is optimized for internal use in the analyzer, several I/O modes are implemented to facilitate the data transfer.

This application note gives examples on how to use the IEC-interface with an HP9825 desktop calculator and also some hints for the use of the analyzer in general.

A few new memory locations not mentioned in the Instruction Manual are also described.

Select Code and Device Address
The select code 7 and device address decimal 25 is used throughout all the programming examples.

I/O Modes
The 2033 has 10 I/O modes. The response of the 2033 to each of these modes is given in Table 1.

2033 Interface mode syntax
The following syntax is used when data are sent to the analyzer:

#m <, c <, d >>, <DATA>

When data are to be dumped from the analyzer the following syntax is used:

#m <, c <, d >>;

where:

m = Mode number
c = Alpha numeric ASCII string
d = Numeric ASCII string

The items enclosed in < > are optional. Mode #8 and #9 are not used for data transfer and have their own syntax:

#8,

#9,

Mode number sent to 2033	2033 response	
	2033 addressed as Listener	2033 addressed as Talker
# Ø	Digital Input ASCII spectrum into Memory	Digital output of ASCII spectrum selected by DISPLAY SELECTOR
# 1	ASCII setting of pushkeys	Settings of all pushkeys output in ASCII
# 2	Digital input and transformation of time function	Digital output of time function
# 3	Block transfer MSBY-LSBY to 2033	Block transfer MSBY-LSBY from 2033
# 4	ASCII addressing of pushkey to be sensed	ASCII output of sensing information
# 5	–	Digital output of 2's complement Spectrum selected by DISPLAY SELECTOR
# 6	Block transfer LSBY to 2033	Block transfer LSBY from 2033
# 7	Block transfer MSBY to 2033	Block transfer MSBY from 2033
# 8	Carry on Processing	–
# 9	Reset 10 K Time Function Buffer	–

810492

Table 1. Digital input/output modes

Delimiters

When the programming sequence for output of data is sent the semicolon is used as delimiter, but when mode # 8 or # 9 are sent comma is used as delimiter.

When data is received from the analyzer the last byte is the ETX with the EOI line true.

If the ETX or the last part of the data are not read by the computer the 2033 will stop processing, but will return to normal operation when the interface clear message is sent (cli7).

Timing

The IEC-interface itself does not give rise to any timing problems, but the following time constants should be observed for the analyzer:

1) 1 K transform and averaging: 200 ms
2) 10 K transform and averaging: 1500 ms
3) Pushkey setting: 25 ms

When the analyzer is in recording stop a transform is performed when the Inst. spectrum is selected or the zoom pushkey is activated.

Memory Organisation

The organisation of the 2033 memory is given in Table 2. Some of the memory locations are not described in the Table since they are for internal use only.

Decimal Address	Memory Contents
0 : 1023	1 K Time buffer
1024 : 2049	Complex spectrum
2050 : 2849	Averaged spectrum
2855	Interrupt mask
2872 : 2922	IEC Text Line
3073 : 3472	Reference Memory
3473 : 3477	Stored Pushkeys
3504	No. of Linear avg.
3520 : 3570	2033 Text Line
3582	Scaling exponent for complex spectrum
3591	Text Line selector
3595	Relative time sample
3596	Relative spectrum Level
3696 : 4095	Display Buffer
4096 : 14335	10 K Time Buffer
14336 : 15361	Instantanous spectrum
16382	Time signal start sample

810052

Table 2. Memory organization

REMOTE CONTROL OF THE FRONT PANEL

I/O of Front Panel Pushkeys

Most of the front panel pushkeys can be remotely controlled from the computer:
(See Section 9.3 of Instruction Manual for a complete listing of codes).

Setting of Pushkeys:

```
wtb 725,"#1,L1;"
```

More than one button can be set using the same statement:

```
wtb 725,"#1,L1,K0;"
```

Sensing of Pushkeys:

```
wtb 725,"#4,L;"
red 725,A
```

where
A = Pushkey code number

All the pushkeys can be sensed at the same time:

```
0: dim A$[232]
1: wtb 725,"#1;"
2: fmt 1,z,c232
3: red 725.1,A$
```

where
A$ = Pushkey code string

I/O of Stored Pushkey Settings

Six pushkey settings are stored in five memory locations (2 of the pushkey settings are stored as a sum) after the Reference spectrum. The address is given in Table 3.

Inp. Att. + Ref. adj. code	3473
Full scale frequency code	3474
Samples a. trig. code	3475
No. of avg. *	3476
Centre freq. code **	3477

* The value is the actual no. of avg.
** The value is Ø if a baseband spectrum is stored.

810049

Table 3. Address of stored pushkeys

Setting of Stored Pushkeys:

```
fti (A)→A$
wtb 725,"#3,3473,1,",A$
```

Where:
A$ = Stored pushkey string code
A = Stored pushkey code

Sensing of Stored Pushkeys:

```
wtb 725,"#3,3473,1;"
fmt 1,z,c2,b
red 725.1,A$,B
itf(A$)→A
```

Where:
A$ = Stored pushkey string code
A = Stored pushkey code
B = Dummy

Dump of Text Line Information

Often it is easier to read the text line information than to calculate the same information using the Pushkey codes. The Text line has 7 fields (6 when a time function is displayed). The start address of each field is given in Table 4.

```
wtb 725,"#6,3522,6;"
red 725,A
(cli 7 or cmd 7,"_")
```

Where:
A = Full scale level for spectra

		Start address	Number of bytes
Spectrum	Full scale level*	3522	6
	Full scale freq. or span*	3529	3
	Samples after trig.	3535	4
	No. of spectra	3540	4
	Selected line	3546	4
	Selected freq.	3551	7
	Selected level	3561	8
Time Signal	Full scale level	3522	4
	Full scale freq.*	3529	3
	Samples after trig	3535	4
	No. of spectra	3540	5
	Selected line	3545	5
	Selected time	3551	7

* Only the numeric value of the field is read i.e. information about the unit is not included.
It is recommended to send Interface clear or UNTALK (cli 7 or cmd 7, "–") after the field has been read, otherwise the analyzer might stop processing.

810053

Table 4. Start address of text line fields

I/O of Alphanumeric Text

Dump from 2033 text line:

```
dim A$[49]
wtb 725,"#6,2873,49;"
fmt 1,z,c49
red 725.1,A$
end
```

Dump from IEC text line:

```
dim A$[49]
wtb 725,"#6,3521,49;"
fmt 1,z,c49
red 725.1,A$
end
```

Load of text into the IEC text line

```
dim A$[51]
" "→A$
for I=2 to 50
"A"→A$[I]
next I
" "→A$[51]
wtb 725,"#6,3520,51," ,A$
end
```

Note that the first and the last character in the text line is only partly displayed and should therefore be filled out with a space.

Update of IEC text line:

```
wtb 725,"#7,3591,1,I"
```

Update of 2033 text line:

```
wtb 725,"#7,3591,1,A"
```

Setting of Reference Time and Level

The reference used when the relative pushkey is activated can be changed from the calculator.

Time reference setting:

```
wtb 725,"#3,3595,1,",fti (A)
```

Where:
A = Sample no. for rel. time 0

Level reference setting:

```
wtb 725,"#3,3596,1,",fti (A)
```

where
A = 10 × (dB level of the reference)

Pushkey Code Conversions for Input Spectra

Full Scale Frequency in Baseband Mode:

```
((Amod3)↑2+1)*10↑(int(A/3)+1)→F
int(3*log(F)-2.5)→A
```

Where:
A = Full scale frequency code
F = Full scale frequency of baseband (Hz)

Frequency Span in Zoom Mode:

```
(10*(B+1)-199)/4000*F→C
(10*(B+1)+200)/4000*F→D
```

Where:
B = Centre frequency code
F = Full scale frequency of baseband (Hz)
C = Lowest displayed frequency (Hz)
D = Highest displayed frequency (Hz)

F.S. FREQUENCY Setting	Code
10 Hz	@ Ø
20 Hz	@ 1
50 Hz	@ 2
100 Hz	@ 3
200 Hz	@ 4
500 Hz	@ 5
1 kHz	@ 6
2 kHz	@ 7
5 kHz	@ 8
10 kHz	@ 9
20 kHz	@1Ø

780162/1

Table 5. F.S. frequency code

Centre Frequency / F.S. FREQUENCY setting	Code
20/400	H 19
21/400	H 2Ø
⋮	⋮
379/400	H 378
380/400	H 379

800140

Table 6. Centre frequency code

ADC Range	Code
± 2,82 mV	D Ø
± 8,91 mV	D 1
± 28,2 mV	D 2
± 89,1 mV	D 3
± 282 mV	D 4
± 891 mV	D 5
± 2,82 V	D 6
± 8,91 V	D 7
± 28,2 V	D 8
± 89,1 V	D 9
± 282 V	D1Ø

780166/1

Table 7. Input Att. code

DISPLAY SELECTOR Setting		Code
"Input Function" "Memory"	I/M	
− 10 dB	− 40 dB	FØ
0 dB	− 30 dB	F1
+ 10 dB	− 20 dB	F2
+ 20 dB	− 10 dB	F3
+ 30 dB	0 dB	F4
+ 40 dB	+ 10 dB	F5
+ 50 dB	+ 20 dB	F6
+ 60 dB	+ 30 dB	F7
+ 70 dB	+ 40 dB	F8

800153

Table 8. Spectrum gain code

REFERENCE ADJUST dB	√	−50 dB	√ Ø
		+ 50 dB	√ 1Ø

810051

Table 9. Reference Adj. code

Full Scale Level:

```
10*(A+B-C+2)→L
10↑(L/20)→M
```

Where:
A = Input Att. code
B = Ref. Adj. code
C = Spectrum gain code
L = Full scale level (dB)
M = Full scale level (μ V)

Bottom Line Level for dB Spectra:

```
L-10*2↑(3-D)→N
```

Where:
L = Full scale level (dB)
D = Spectrum range code
N = Bottom line level (dB)

Number of Spectra:

```
2↑A→B
9*2↑(A-12)+1→C
int(ln(B)/ln(2)+.5)→D
int(ln((B-1)/9)/ln(2)+12+.5)→E
```

Where:
A = Number of spectra code
B = Number of spectra for normal avg.
C = Number of spectra for scan avg.
D = Number of spectra code for normal avg.
E = Number of spectra code for scan avg.

The actual number of spectra averaged can be read from the text line (see Table 4).

"No. of Spectra" setting	Code
1	BØ
2	B1
4	B2
8	B3
16	B4
32	B5
64	B6
128	B7
256	B8
512	B9
1024	B1Ø
2048	B11

"No of Spectra" setting	Code
10	B12
19	B13
37	B14
73	B15
145	B16
289	B17
577	B18
1153	B19

800138

Table 10. No. of spectra code

Stored Pushkey Code Conversions

The same formulae as used for the input spectra can be used for the stored spectra, except that there is no code for No. of spectra and that the sum of the codes for the Input Att. and Ref. Adj. is used directly.

Full Scale Level for I/M Spectra

```
10*(A+B-C-D+8)→L
10↑((L-30)/20)→M
```

Where:
A = Input Att. code
B = Ref. Adj. code
C = Spectrum gain code
D = Input Att. + Ref. Adj. code for stored spectrum
L = Full scale level (dB)
M = Full scale level ratio for linear spectra

Pushkey Code Conversions for Time Signals

1 K Record Length

```
400/A→B
```

Where:
A = Full scale frequency of baseband spectrum (Hz)
B = Record length of 1 K time signal (Sec)

10 K Record Length

```
4000/A→B
```

Where:
A = Full scale frequency of baseband spectrum (Hz)
B = Record length of 10 K time signal (Sec)

390 Line Selected Interval

```
A-194→C
A+195→D
```

Where:
A = Move setting code
B = Line selector code
C = First sample displayed
D = Last sample displayed

First Sample "1 K"	Last Sample "1 K"	First Sample "390"	Last Sample "390"	Code
		1	390	E 195
		2	391	E 196
		⋮	⋮	⋮
1	1024	318	707	E 512
⋮	⋮	⋮	⋮	⋮
9217	10240	9534	9923	E 9728
		⋮	⋮	⋮
		9850	10239	E 1ØØ44
		9851	10240	E 1ØØ45

800139

Table 11. Move setting code

Line Number	Code
1	GØ
2	G1
3	G2
.	.
.	.
.	.
399	G398
400	G399

780169/1

Table 12. Line selector code

1 K Line Selected Interval

```
A-511→C
A+512→D
```

where:
A = Move setting code
B = Line selector code
C = First sample displayed
D = Last sample displayed

Full Scale Level

```
10↑((10*(A+B)+19)/20)→C
```

Where:
A = Input Att. setting code
B = Ref. Adj. setting code
C = Full scale level (μ V)

INPUT/OUTPUT OF POWER SPECTRA

The measured spectrum can be dumped from 4 buffers:

(1) Instantaneous spectrum buffer
(2) Avg. spectrum buffer
(3) Reference memory buffer
(4) Display buffer

The spectrum in (1) is the power spectrum, i.e. the sum of the squares of the real and Imaginary part, the spectrum in (2) is the average of these spectra.

The spectrum in (3) is the dB level stored in Ref. memory and (4) contains the height of the lines displayed on the analyzer screen.

Two other modes can be used to output the spectrum:

(1) Mode # 5
(2) Mode # 0

Mode # 5 outputs the same level as the selected level displayed in the text line, but all 400 lines.

This mode is the most useful since it can dump any spectrum displayed on the screen.

Since the spectral values are calculated during the read-out and not stored in the analyzer, this mode can only be used to output spectra.

Mode # 0 is only used for manual output and for calculators not able to read binary coded data. (See Instruction Manual for more information.)

I/O of dB Spectra

Dump of Selected Spectrum (mode # 5)
The output is 10 times the dB level coded as two's complement.

```
0: dim A$[800+1+16],A[400]
1: buf "IN",A$,3
2: gsb "DUMP-1"
3: gsb "CONVERT-1"
4: end
5:
6: "DUMP-1":
7: wtb 725,"#5;"
8: buf "IN"
9: tfr 725,"IN"
10: jmp rds("IN")#-1
11: ret
12:
13: "CONVERT-1":
14: for I=1 to 400
15: .1*itf(A$[2I-1,2I])→A[I]
16: next I
17: ret
*23462
```

Where:
A [*] = dB spectrum

Dump from Reference Memory + Stored Pushkeys

```
0: dim A$[810+1+16],A[400]
1: buf "IN",A$,3
2: gsb "DUMP-2"
3: gsb "CONVERT-1"
4: end
5:
6: "DUMP-2":
7: wtb 725,"#3,3073,405;"
8: buf "IN"
9: tfr 725,"IN"
10: jmp rds("IN")#-1
11: ret
*29254
```

The output is 10 times the dB level, coded as two's complement followed by the stored pushkey codes.The stored pushkey codes are held in A$ [801, 810].

Load of Spectra into the Reference Memory
Spectra loaded into the Ref. Memory buffer are only displayed if the Store Button has been activated after power-up or system reset and before the spectrum is loaded.

Spectra dumped using "DUMP – 2" are loaded as follows:

```
wtb 725,"#3,3073,405,",A$[1,810]
```

Spectra dumped using "DUMP – 1" can also be loaded into the Reference Memory:

```
wtb 725,"#3,3073,400,",A$[1,800]
```

but the codes for the stored pushkeys must be loaded separately.

Example:
Load of 400 line spectrum of 100 dB into Ref. Memory
Full scale level: 110 dB

Full scale frequency: 10 kHz in baseband
Samples after trig.: 5
No. of avg.: 10
Zoom center frequency 2500 Hz

```
0: dim A$[800]
1: wtb 725,"#1,M1;"
2: for I=1 to 400
3: fti (10*100)→A$[2I-1,2I]
4: next I
5: wtb 725,"#3,3073,400,",A$
6: wtb 725,"#3,3473,5,",fti (10),fti (9),fti (50),fti (10),fti (99)
7: end
*10372
```

Dump from Instantaneous and Average Buffer

The format of the output of the two buffers is the same with the following two exceptions:

1) The DC line and line 401 - 512 are not available for the avg. spectrum, but since the analyzer is AC-coupled and since the anti-aliasing filter cuts off at line 400, these lines are usually of no interest.

2) If the avg. spectrum has been measured using lin. avg., the data in the avg. buffer must be divided by the actual no. of spectra. The actual no. of spectra can be read from the text line or by using:

```
wtb 725,"#3,3504,1;"
fmt 1,z,b,b,b
red 725.1,A,B,C
256*A+B→C
```

Where:
C = Actual no. of avg. (C = 1 for exp. avg.)

The output is the sum of the squares of the Real and Imag. part, coded as a mantissa and an exponent, the mantissa being in binary format the exponent in two's complement binary format.

Dump of 400 Line Inst. Power Spectrum

```
0: dim A$[1600+1+16],A[400]
1: buf "IN",A$,3
2: gsb "DUMP-3"
3: gsb "CAL-3"
4: gsb "CONVERT-3"
5: end
6:
7: "DUMP-3":
8: wtb 725,"#3,14338,800;"
9: buf "IN"
10: tfr 725,"IN"
11: jmp rds("IN")#-1
12: ret
13:
14: "CAL-3":
15: wtb 725,"#4,D;"
16: red 725,A
17: wtb 725,"#4,√;"
18: red 725,B
19: 10*(A+B-2)→S
20: ret
21:
```

```
22: "CONVERT-3":
23: 1/65536→K
24: for I=1 to 400
25: gsb "CONV-1-LINE"
26: 10*log(X)+S→A[I]
27: next I
28: ret
29:
30: "CONV-1-LINE":
31: 4*I-3→r1
32: 1+K*itf(A$[r1,r1+1])→r2
33: itf(A$[r1+2,r1+3])→r3
34: r2*2↑r3→X
35: ret
*15937
```

Dump of Avg. Power Spectrum

The following two sub-routines are used instead of DUMP-3 and CAL-3.

```
0: "DUMP-4":
1: wtb 725,"#3,2050,800;"
2: buf "IN"
3: tfr 725,"IN"
4: jmp rds("IN")#-1
5: ret
6:
7: "CAL-4":
8: wtb 725,"#4,D;"
9: red 725,A
10: wtb 725,"#4,√;"
11: red 725,B
12: wtb 725,"#3,3504,1;"
13: fmt 1,z,b,b,b
14: red 725.1,A,B,D
15: 256*A+B→C
16: 10*(A+B-2-log(C))→S
17: ret
*13163
```

Load of Spectra into the Instantaneous or Average Buffer

Spectra dumped using the sub-routines DUMP-3 or DUMP-4 can be loaded directly into one of the two buffer:

Inst. buffer:

```
wtb 725,"#3,14338,800,",A$[1,1600]
```

Avg. buffer:

```
wtb 725,"#3,2050,800,",A$[1,1600]
wtb 725,"#3,3504,1,",fti (A)
```

Where:

$$A = \begin{cases} \text{1 for Inst. and exp. avg. spectra} \\ \text{Actual no. of spectra for Lin. avg. spectra} \end{cases}$$

Example:

Full scale level 110 dB.
Load of 400 line spectrum of 100 dB into avg. buffer.

```
0: dim A$[1600]
1: 1/log(2)→K
2: 5+5-2→S
3: for I=1 to 400
4: .1*100-S→L
5: int(K*L)+1→E
6: int(65536*(2↑(-E)*tn↑L-1))→M
7: if M=0;-1→M
8: if M<-32768;-32768→M
9: fti (M)→A$[4I-3,4I-2]
10: fti (E)→A$[4I-1,4I]
11: next I
12: wtb 725,"#1,[1,K0,L1,D5,F1;"
13: wtb 725,"#3,3504,1,",fti (1)
14: wtb 725,"#3,2050,800,",A$[1,1600]
15: end
*11146
```

Note: The Ref. adj. must be set to 0 dB.

Dump from Display Buffer

The output is the height of the lines displayed on the analyzer screen coded in binary format.The bottom line being 28 and the full scale line 228.

```
0: dim A$[400+1+16],A[400]
1: buf "IN",A$,3
2: gsb "DUMP-5"
3: gsb "CONVERT-5"
4: end
5:
6: "DUMP-5":
7: wtb 725,"#6,3696,400;"
8: buf "IN"
9: tfr 725,"IN"
10: jmp rds("IN")#-1
11: ret
12:
13: "CONVERT-5":
14: for I=1 to 400
15: num(A$[I,I])→A[I]
16: next I
17: ret
*1750
```

Where:
A[*] = Height of displayed lines.

Since the display is continuously updated only output of spectra is possible.

INPUT/OUTPUT OF TIME SIGNALS

The measured Time Signal can be dumped from 3 buffers:

(1) 10 K time buffer
(2) 1 K time buffer
(3) Display buffer

The time signal in (1) is the output from the ADC and is only available when the analyzer is in recording stop.

The time signal in (2) is the 1 K time signal that is transformed. The data are scaled down by a factor of 2 relative to the data in (1).

(3) contains the height of the lines displayed on the analyzer screen measured from the bottom line.

I/O of the 10 K Time Buffer

Input and output is only possible when a 1 K or 10 K time signal is displayed and only when the analyzer is in recording stop.

The data are formated as two's complement. The bottom line being – 32768 and the top line being 32767.

The data stored in the 10 K time buffer can be output using mode # 2 or mode # 3, but as the starting point of the 10 K record can be any sample, mode # 2 is recommended since the data are automatically output in the right order.

Dump using Mode # 2:
The output is 16 characters giving information about overload and record length (see Instruction Manual) followed by the time signal data. When a 1 K time signal is displayed the output is a 1 K time record, when the 10 K time signal is displayed the output is the 10 K time signal.

Dump of 1 K Time Signals

```
0: dim A$[16+2048+1+16],A[1024]
1: buf "IN",A$,3
2: gsb "DUMP-6"
3: gsb "CAL-6"
4: gsb "CONVERT-6"
5: end
6:
7: "DUMP-6":
8: wtb 725,"#2;"
9: buf "IN"
10: tfr 725,"IN"
11: jmp rds("IN")#-1
12: ret
13:
14: "CAL-6":
15: wtb 725,"#4,D;"
16: red 725,A
17: wtb 725,"#4,√;"
18: red 725,B
19: 10↑((10*(A+B)+19)/20)→S
20: ret
21:
22: "CONVERT-6":
23: S/32768*1e-6→K
24: for I=1 to 1024
25: K*itf(A$[2*(I+8)-1,2*(I+8)])→A[I]
26: next I
27: ret
*18474
```

Where:
A[*] = Level in Volts

Dump of 10 K Time Signal

```
0: dim A$[16+20480+1+16]
1: buf "IN",A$,3
2: gsb "DUMP-6"
3: end
*12793
```

Since the 10 K time record uses a large part of the calculator memory it is usually more convenient to dump ten 1 K records and store each on tape.

```
0: dim A$[16+2048+1+16],B$[2048]
1: buf "IN",A$,3
2: for I=1 to 10
3: fxd 0;wtb 725,"#1,E",str(512+1024*(I-1)),";"
4: wait 25
5: gsb "DUMP-6"
6: A$[17,2064]→B$[1,2048]
7: rcf I,B$
8: next I
9: end
*15604
```

Dump using Mode # 3

Using mode # 2 only 1 K or 10 K records can be dumped, using mode # 3 any part of the 10 K time record can be accessed, but the starting point has to be read separately.

```
0: dim A$[20+1+16],B$[20],A[20]
1: buf "IN",A$,3
2: 1000→M
3: 10→L
4: wtb 725,"#3,16382,1;"
5: fmt 1,z,b,b,b
6: red 725.1,A,B,C
7: 256*A+B→A
8: (A+M-1-4096)mod10240+4096→S
9: (A+M+L-2-4096)mod10240+4096→T
10: if S>T;gto "CASE-II"
11: "CASE-I":
12: fxd 0;wtb 725,"#3,",str(S),",",str(L),";"
13: gsb "DUMP-7"
14: A$[1,2L]→B$[1,2L]
15: gto "CONV"
16: "CASE-II":
17: 14336-S→U
18: L-U→V
19: fxd 0;wtb 725,"#3,",str(S),",",str(U),";"
20: gsb "DUMP-7"
21: A$[1,2U]→B$[1,2U]
22: fxd 0;wtb 725,"#3,4096,",str(V),";"
23: gsb "DUMP-7"
24: A$[1,2V]→B$[2U+1,2L]
25: "CONV":
26: gsb "CAL-7"
27: gsb "CONVERT-7"
28: end
29:
```

```
30: "DUMP-7":
31: buf "IN"
32: tfr 725,"IN"
33: jmp rds("IN")#-1
34: ret
35:
36: "CAL-7":
37: wtb 725,"#4,D;"
38: red 725,D
39: wtb 725,"#4,√;"
40: red 725,E
41: 10↑((10*(D+E)+19)/20)→F
42: ret
43:
44: "CONVERT-7":
45: F/32768*1e-6→K
46: for I=1 to L
47: K*itf(B$[2I-1,2I])→A[I]
48: next I
49: ret
*5262
```

Load into 10 K Time Buffer (Mode # 2)
1 K time signals dumped using DUMP-6 are loaded as follows:

```
wtb 725,"#9,#2,",A$[1,2064]
```

10 K time signals dumped using DUMP-6 are loaded as follows:

```
wtb 725,"#9,#2,",A$[1,20496]
```

Example
Load of a 1 K sinusoid into the 10 K time buffer:

```
0: dim A$[2048]
1: for I=1 to 1024
2: fti (10000*sin(360/1024*I*5))→A$[2I-1,2I]
3: next I
4: wtb 725,"#9,#2,16,1,",A$[1,2048]
5: end
*7268
```

Load into 10 K Time Buffer (mode # 3)
Before a time signal is loaded into the 10 K time buffer using mode # 3 the start address of the buffer must be read. After the signal has been loaded only the 390 sample display is updated. In order to update the 1 K sample display, the Time function Move pushkey must be changed. In order to update the 10 K sample display the zoom pushkey must be activated.

Example
Load of 10 lines of equal level into the 10 K time buffer:

```
0: dim A$[20]
1: 1000→M
2: 10→L
3: for I=1 to 10
4: fti (10000)→A$[2I-1,2I]
5: next I
6: wtb 725,"#3,16382,1;"
7: fmt 1,z,b,b,b
8: red 725.1,A,B,C
9: 256*A+B→A
10: (A+M-1-4096)mod10240+4096→S
11: (A+M+L-2-4096)mod10240+4096→T
12: if S>T;gto "CASE-II"
13: "CASE-I":
14: fxd 0;wtb 725,"#3,",str(S),",",str(L),",",A$[1,20]
15: gto "UPDATE"
16: "CASE-II":
17: 14336-S→U
18: L-U→V
19: fxd 0;wtb 725,"#3,",str(S),",",str(U),",",A$[1,2U]
20: wtb 725,"#3,4096,",str(V),",",A$[2U+1,2L]
21: "UPDATE":
22: wtb 725,"#1,E512;"
23: wtb 725,"#1,E1000;"
24: end
*12566
```

Loading of time signals into the 1 K time buffer or the display buffer is not recommended since the contents of these buffer are changed when the Move button or other display control buttons are activated.

Dump from 1 K Time Buffer

When the analyzer is in continuous recording and internal or external trigger is used, the 10 K time record is continuously updated although the display is not changing, therefore data cannot be dumped from the 10 K time buffer. But when a 1 K time signal is displayed the data can be dumped from the 1 K time buffer. Only mode # 3 can be used to dump the data. The data are formated as two's complement. The bottom line being – 16384 and the top line being 16383.

```
0: dim A$[2048+1+16],A[1024]
1: buf "IN",A$,3
2: gsb "DUMP-8"
3: gsb "CAL-6"
4: gsb "CONVERT-8"
5: end
6:
7: "DUMP-8":
8: wtb 725,"#3,0,1024;"
9: buf "IN"
10: tfr 725,"IN"
11: jmp rds("IN")#-1
12: ret
13:
```

```
14: "CONVERT-8":
15: S/16384*1e-6→K
16: for I=1 to 1024
17: K*itf(A$[2I-1,2I])→A[I]
18: next I
19: ret
*12434
```

Where:
A[*] = Level in volts

Dump from Display Buffer

The output is the height of the lines displayed on the analyzer screen, measured from the bottom line. The bottom line being 28 and the top line being 228.

(See example in section: Dump from Spectrum Display Buffer)

OUTPUT OF COMPLEX SPECTRA

The complex spectrum can be dumped from 2 buffers:

(1) Complex spectrum of 512 sample complex signal
(2) Complex spectrum of 1024 sample real signal

The complex spectrum in (1) is the Fourier transform of a complex time signal, where the time signal is loaded so that the even numbered samples correspond to the real part and the odd numbered samples correspond to the imaginary part (see Instruction Manual for further information).

The complex spectrum in (2) is the Fourier transform of a real time signal. As the real part is an even function and the Imag. part is an odd function only the values for positive frequencies are stored. The common scaling exponent stored in the memory must be read after the spectrum is dumped, in order to obtain the correct scaling.

Dump from Complex Buffer (2)

The following two conditions must be satisfied in order to dump the spectrum:

1) Bit 3 of the interrupt mask is set
2) The Fourier transform is initiated

Condition 2 can be obtained as follows:

Select Input function Inst.

and

1) Cont. or Single Recording

or

2) Recording Stop

and

a) Load of time signal using mode # 2

or

b) Activate Time Function Move if not Zoom

or

c) Activate centre freq. if Zoom

The scaling of the complex spectrum is:

2 ↑ (S + 18)

Where:
S = Value of memory location 3582

```
0: dim A$[2052+1+16],R[513],I[513]
1: buf "IN",A$,3
2: oni 7,"INTR"
3: gsb "DUMP-9"
4: gsb "CONVERT-9"
5: end
6:
7: "DUMP-9":
8: cfg 1
9: wtb 725,"#6,2355,1,",4
10: wtb 725,"#1,X2;"
11: eir 7
12: eir 7,0
13: if not flg1;jmp -2
14: wtb 725,"#6,2855,1,",0
15: cfg 1
16: ret
17:
18: "INTR":
19: rds(7)→A
20: if not bit(7,A);iret
21: rds(725)→A;cli 7
22: if not bit(6,A);iret
23: wtb 725,"#3,1024,1026;"
24: buf "IN"
25: tfr 725,"IN"
26: jmp rds("IN")#-1
27: sfg 1
28: wtb 725,"#3,3582,1;"
29: fmt 1,z,b,b,b
30: red 725.1,A,B,C
31: 256*A+B-65536*(A>127)→S
32: iret
33:
34: "CONVERT-9":
35: 2↑(S+18)→S
36: for I=1 to 513
37: S*itf(A$[4I-3,4I-2])→R[I]
38: S*itf(A$[4I-1,4I])→I[I]
39: next I
40: ret
*4423
```

Bruel & Kjaer Instruments, Inc.
185 Forest Street, Marlboro, Massachusetts 01752 / (617) 481-7000