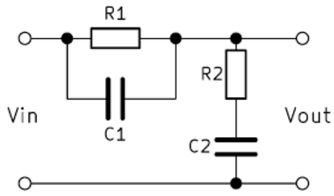
HP-41 EE Filters Module



Overview

This module comprises a selection of programs and functions loosely grouped around the EE Filters theme. It's meant to complement the HP EE Solutions Book, notably the hp-41 version of the Butterworth & Chebyshev's Filters program published by T. Mickelson in PPCCJ V7N6. Other programs have been taken from the PRISMA magazine, the official publication of the CCD German user's Club.Finally a section in the upper page includes the programs by Dieter Lange from the Vieweg book "Electrical Networks Analysis"

The table below shows the function names in alphabetical order with a brief description. The Authors and sources are listed below for a complete program description and user instructions in case you're interested.

XROM	Function	Description	Author	Source
17,00	**-OPAMP 41**	*Section Header*	*Ángel Martin*	*This project*
17,01	**AGM**	Arithmetic-Geometric Mean	*Ángel Martin*	SandMath project
17,02	**AGM2**	Extended AGM	*Ángel Martin*	Paper by
17,03	**ELIPE**	Elliptic Integral 2nd Order	*Ángel Martin*	This project
17.04	**ELIPK**	Elliptic Integral 1st Order	*Ángel Martin*	This project
17.05	**"EK"**	Elliptic Integral 2nd Order	*Ángel Martin*	This project
17.06	**"KK"**	Elliptic Integral 1st Order	*Ángel Martin*	This project
17.07	**"MIND"**	Mutual Inductance	*Ángel Martin*	*NASA SP-42*
17.08	**"NOIZ"**	Noise Factor	*Unknown?*	*Best of Prisma*
17.09	**"OPGOFF"**	OpAmp Gain and Offset	*Stefan Vorkoetter*	*Author's Web*
17.10	**"OPOSC"**	OpAmp Oscillators	*Stefan Vorkoetter*	*Author's Web*
17.11	**"SLNKEY"**	Sallen-Key Filter Design	*Stefan Vorkoetter*	*Author's Web*
17.12	**"="**	Builds Prompt in ALPHA	*Ángel Martin*	*This project*
17.13	**-41 FILTERS**	*Section Header*	*Ángel Martin*	*This project*
17.14	**"BC"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.15	**"C3"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.16	**"C?"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.17	**"FB"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.18	**"N"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.19	**"O"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.20	**"T"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.21	**"Z"**	BW & Chebyshev Filters	*Terry Mickelson*	*PPCCJ V9N2p20*
17.22	**dB-**	Decibel Subtraction	*Ángel Martin*	*ETSII Project*
17.23	**dB+**	Decibel Addition	*Ángel Martin*	*ETSII Project*
17.24	**-PRISMA EE**	*Section Header*	*Ángel Martin*	*This project*
17.25	**ACOSH**	Inverse Hyperbolic COS	*Ángel Martin*	*SandMath project*
17.26	**ASINH**	Inverse Hyperbolic SIN	*Ángel Martin*	*SandMath project*
17.27	**"A2HP"**	Active High-Pass 2nd order	*Heinrich Henze*	*Best of Prisma*
17.28	**"A2LP"**	Active Low-Pass 2nd order	*Heinrich Henze*	*Best of Prisma*

17,29	**"APPROX"**	Digital Filter Approximation	*Thomas Wegmann*	*Prisma 1985.V3*
17,30	**“BWHP”**	Butterworth High-Pass Filter	*Werner Meschede*	Prisma 1988 V2
17,31	**“BWLP”**	Butterworth Low-pass Filter	*Werner Meschede*	Prisma 1988 V2
17,32	**“CBHP”**	Chebyshev High-Pass Filter	*Werner Meschede*	Prisma1987 V6
17,33	**“CBLP”**	Chebyshev Low-Pass Filter	*Werner Meschede*	Prisma1987 V6
17,34	**“CBPB”**	Chebyshev Pass-band filter	*Werner Meschede*	Prisma 1988 V1
17,35	**“CBSP”**	Chebyshev Stop-band filter	*Werner Meschede*	Prisma 1988 V1
17,36	**“END”**	Aux. routine	*Werner Meschede*	Prisma V2 1988
17.37	**“SHOW”**	Aux routine	*Werner Meschede*	Prisma V2 1988
18,00	**-NETWRK 41**	*Section Header*	*Ángel Martin*	*This project*
18,01	**“INET”**	Current Nodes	Dieter Lange	*Network Analysis*
18,02	**“UNET”**	Voltage Nodes	Dieter Lange	*Network Analysis*
18,03	**“U0”**	Input Voltage	Dieter Lange	*Network Analysis*
18,04	**“I0”**	Input Current	Dieter Lange	*Network Analysis*
18,05	**“<)0”**	Input Lag Angle	Dieter Lange	*Network Analysis*
18,06	**“BN”**	Bode Network	Dieter Lange	*Network Analysis*
18,07	**“LIST”**	List Elements	Dieter Lange	*Network Analysis*
18,08	**“C*”**	Complex Product	Dieter Lange	*Network Analysis*
18,09	**“*BD”**	Aux. routine	Dieter Lange	*Network Analysis*
18,10	**“*LST”**	Aux routine	Dieter Lange	*Network Analysis*
18,11	**“RED”**		Dieter Lange	*Network Analysis*
18,12	**“U1”**	Output Voltage	Dieter Lange	*Network Analysis*
18,32	**“I1”**	Output Current	Dieter Lange	*Network Analysis*
18,14	**“<)1”**	Output Lag Angle	Dieter Lange	*Network Analysis*
18,15	**“BR”**		Dieter Lange	*Network Analysis*
18,16	**-NETWRK FR**	*Section Header*	*Ángel Martin*	*This project*
18,17	**“NETFR”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,18	**“LIN”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,19	**“LOG”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,20	**“R”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,21	**“L”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,22	**“C”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,23	**“VCS”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,24	**“R=”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,25	**“L=”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
16,26	**“C=”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,27	**“I<V>=”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,28	**“-C”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,29	**“*C”**	Network Frequency Response	Michael Moser	*UPLE# 10786*
18,30	**“/C”**	Network Frequency Response	Michael Moser	*UPLE# 10786*

A few Auxiliary Functions

Amongst the new contents, MCODE functions to calculate the Elliptic Integrals of first and second order are also included, together with one example of usage (Mutual inductance between 2 coaxial coils). Note that these functions are based on the Arithmetic-Geometric Mean and its extension, resulting in a much faster execution than the traditional approach based on Hypergeometric Functions.

Below is a brief description for some of the functions of the module for your convenience – You're encouraged to check the documentation provided in the linked documents for all the other functions and programs; in particular Stefan Vorkoetter's pages are world-class documentation worth checking out.

Hyperbolic Functions. { **ACOSH**, **ASINH** }

These functions calculate the main values of the argument of the hyperbolic sine and cosine, as defined by the well-known relationships:

asinh (x) = Ln [x + SQRT(x^2 +1)]

acosh (x) = Ln [x + SQRT(x^2 – 1)]

They are used in the Chebyshev's filter design by approximation (program "APPROX")

Arithmetic-Geometric Means {**AGM**, **AGM2** }

These functions calculate the standard and extended arithmetic-geometric means of two values, stored in the X and Y registers.

The result is left in the X register, and the second argument (originally in X) is saved in LastX.

STACK	INPUT/OUTPUT	OUTPUT/INPUT
Y	y	z
X	x	AGM(y,x) / AGM2(y,x)
L	-	x

Example1: Calculate the AGM of 17 and 23:

17, ENTER^, 23, XEQ "AGM" => 19.88669905

Example 2: Calculate the extended AGM of 1 and 2:

1, ENTER^, 2, XEQ "AGM2" => 1,456946581

Complete Elliptic Integrals. { ELIPK , ELIPE }

These functions calculate the Elliptic Integrals of first and second orders; K(x) and E(x) respectively. The algorithms are based on the mathematical relationships involving the arithmetic-geometric means, which provide a much faster and slightly more accurate approach than those using hyper-geometric functions – a double-bonus for a 41 implementation. No data registers are used but contents of Alpha registers {O,P} are lost.

	INPUT/OUTPUT	OUTPUT/INPUT
Y	y	y
X	x	E(x) / K(x)
L	-	x

$$E(m) = \int_0^{\pi/2} \sqrt{1 - m \sin^2 \theta}\, d\theta$$

$$K(m) = \int_0^{\pi/2} d\theta / \sqrt{1 - m \sin^2 \theta}$$

Note that the argument x must be ***positive and less than or equal to 1*** – otherwise a DATA ERROR condition will occur.

Example: calculate the difference between the values of both integrals for the argument x=0.5

0.5, XEQ "ELIPK"	=>	1,854074677
LASTX, XEQ "ELIPE"	=>	1,350643881
-	=>	0,503430796

Complete Elliptic Integrals. { EK , KK }

These two are FOCAL counterparts of the MCODE functions described above. Besides relative comparison purposes, they are included for you to see the actual coding of the functions, showing the usage of the AGM functions. The program is very short, and is listed below:

Line	Instruction	Line	Instruction	Line	Instruction
01	**LBL "KK"**	08	**AGM**	15	**XROM "KK"**
02	CHS	09	ST+ X	16	RCL O
03	E	10	1/X	17	X^2
04	+	11	PI	18	E
05	SQRT	12	*	19	**AGM2**
06	STO O	13	RTN	20	*
07	E	14	**LBL "EK"**	21	END

Contrary to their MCODE counterparts, these two will not preserve the stack nor will they save the original argument into the LastX register.

The magic formulas used are as follows – note the (pesky) convention on the argument notation, which is *not* squared:

$$K(x) = \pi / \{2\ AGM\ [1, (1-x)]\}$$

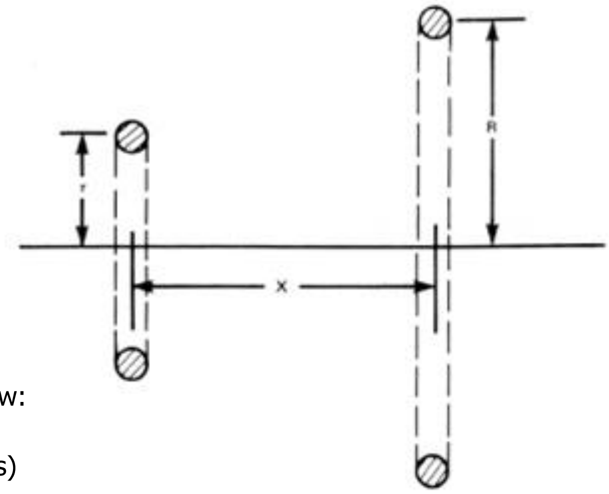
$$E(x) = K(x) * AGM2\ [1, (1-x)^2]$$

Example: Mutual inductance of coaxial coils

This short program perfectly showcases the usefulness of the elliptic integral functions described above. Use it to calculate the mutual inductance between two coaxial coils of radius R1 and R2, separated a distance "d". As it's known, the formula for the mutual induction (in Henries) can be expressed as:

M(R,r,x) $= \frac{8\pi \times 10^{-7}\sqrt{rR}}{k}\left[\left(1 - \frac{k^2}{2}\right)K - E\right]$

Where: m = $k^2 = m = \frac{4rR}{[(R+r)^2 + x^2]}$



Example: Calculate the mutual inductance for the case below:

R1=0.2; R2=0,25; and d=0.1 (all distances in meters)

MIND = 2,48787 E-07 henries

```
01  LBL "MIND"
02  "R1=?"
03  PROMPT
04  STO 01
05  "R2=?"
06  PROMPT
07  STO 02
08  LBL 00
09  "d=?"
10  PROMPT
11  LBL C
12  STO 00
13  RCL 02
14  RCL01
15  *
16  4
17  *
18  RCL 01
19  RCL 02
20  +
21  X^2
22  RCL 00
23  X^2
24  +
25  /
26  STO 00
27  ELIPK
28  STO 03
29  RCL 00
30  ELIPE
31  STO 04
32  E
33  RCL 00
34  2
35  /
36  -
37  RCL 03
38  *
39  RCL 04
40  -
41  PI
42  *
43  8 E-7
44  *
45  RCL 01
46  RCL 02
47  *
48  RCL 00
49  /
50  SQRT
51  *
52  "MI="
53  ARCL X
54  PROMPT
55  GTO 00
56  END
```

Decibel Arithmetic

The decibel (dB) is a logarithmic unit used to express the ratio between two values of a physical quantity, often power or intensity. One of these quantities is often a reference value, and in this case the decibel can be used to express the absolute level of the physical quantity, as in the case of sound pressure. The number of decibels is ten times the logarithm to base 10 of the ratio of two power quantities,[1] or of the ratio of the squares of two field amplitude quantities [2]. One decibel is one tenth of one bel, named in honor of Alexander Graham Bell.

[
1] Power quantities

When referring to measurements of power or intensity, a ratio can be expressed in decibels by evaluating ten times the base-10 logarithm of the ratio of the measured quantity to the reference level. Thus, the ratio of a power value P1 to another power value P0 is represented by LdB, that ratio expressed in decibels,[19] which is calculated using the formula below:

$$L_{\mathrm{dB}} = 10\log_{10}\left(\frac{P_1}{P_0}\right)$$

[2] Field quantities

When referring to measurements of field amplitude, it is usual to consider the ratio of the squares of A1 (measured amplitude) and A0 (reference amplitude). This is because in most applications power is proportional to the square of amplitude, and it is desirable for the two decibel formulations to give the same result in such typical cases. Thus, the following definition is used:

$$L_{\mathrm{dB}} = 10\log_{10}\left(\frac{A_1^2}{A_0^2}\right) = 20\log_{10}\left(\frac{A_1}{A_0}\right).$$

At any rate, the expression included in the module uses the 10-factor for the calculation, not the 20-factor.

The functions **dB+** and **dB-** calculate the result of adding or subtracting two values in X and Y expressed in decibels. The result is also a dB value. You can also use dB+ with a negative sign in X for subtractions.

Examples: 3 dB + 5 dB = 7.124426028 dB
5 dB – 3 dB = 0.670765667 dB

MCODE listing:

The two values are expected in stack registers X, Y
Result is place din X and the former value is saved in LastX

Header	AFC9	0AB	"+"	
Header	AFCA	002	"B"	***Decibel addition***
Header	AFCB	044	"d"	
dB+	**AFCC**	**244**	**CLRF 9**	*Ángel Martin*
	AFCD	02B	*JNC +05*	*[MAIN]*
Header	AFCE	0AD	"-"	
Header	AFCF	002	"B"	***Decibel subtraction***
Header	AFD0	044	"d"	
dB-	**AFD1**	**248**	**SETF 9**	*Ángel Martin*
MAIN	AFD2	0F8	READ 3(X)	
	AFD3	361	*?NC XQ*	*(includes SETDEC)*
	AFD4	050	*->14D8*	*[CHK_NO_S]*
	AFD5	266	C=C-1 S&X	*/10*
	AFD6	070	N=C ALL	
	AFD7	044	CLRF 4	
	AFD8	3E1	*?NC XQ*	
	AFD9	06C	*->1BF8*	*[10TOX]*
	AFDA	24C	?FSET 9	
	AFDB	01B	*JNC +03*	*[PLUS]*
MINUS	AFDC	2BE	C=-C-1 MS	*sign change*
	AFDD	11E	A=C MS	*ditto for 13-digit form*
PLUS	AFDE	089	*?NC XQ*	
	AFDF	064	*->1922*	*[STSCR]*
	AFE0	0B8	READ 2(Y)	
	AFE1	361	*?NC XQ*	*(includes SETDEC)*
	AFE2	050	*->14D8*	*[CHK_NO_S]*
	AFE3	266	C=C-1 S&X	*/10*
	AFE4	070	N=C ALL	
	AFE5	044	CLRF 4	
	AFE6	3E1	*?NC XQ*	
	AFE7	06C	*->1BF8*	*[10TOX]*
	AFE8	0D1	*?NC XQ*	
	AFE9	064	*->1934*	*[RCSCR]*
	AFEA	031	*?NC XQ*	
	AFEB	060	*->180C*	*[AD2-13]*
	AFEC	01E	A=0 MS	*absolute value*
	AFED	088	SETF 5	*Decimal Log*
	AFEE	121	*?NC XQ*	
	AFEF	06C	*->1B48*	*[LN13]*
	AFF0	226	C=C+1 S&X	*x10*
	AFF1	000	NOP	*let carry settle*
	AFF2	369	*?NC GO*	*Final checks & Exit*
	AFF3	002	*->00DA*	*[NFRXY]*

Note the use of 13-digit math routines from the OS, for better accuracy results.

41C Butterworth and Chebyshev Filters.

This is a translation / combination of some of Bruce Murdock's hp-97 programs as they appear in books 1&2 of the HP user's series on Butterworth and Chebyshev filters. The HP-41 program will do all 8 types of bandpass filters, both types of band-stop, both high-pass and both low-pass.

Included under LBL "O" is a routine taken from the hp67 EE PAC. It's supposed to give "n", or the number of elements needed for a filter, given some out-of-band and frequency, and how far down this is to be in dB. It dumps the result into "N" and if it exceeds 9, an error message comes up (implemented in line .54, as TONE IND 03 that only goes to 9). You can change this if more than 9 elements are being run off – but then also need to change line .186 from 23.014 to 25.04 if 11 elements are needed. There are 4 registers left over if two of the single modules are used, so there is room for some changes in the printout or the quantity of elements being done.

Some lines have "illegal" text characters such as .247 (#) and .217 (appends ohms). Lines .278 and .284 have a lower case "z" in MHz but these were read off by using the wand and can be changed. There are no illegal instructions in the program and no card reader functions are used. The printer is necessary.

There are some minor differences between this and Bruce's programs in the order of data entry, and all frequencies are entered in MHz instead of Hz. Some tricks were used to get the high-pass, lowpass and band-stop types into the same label (LBL "T") but all this will be explained.

Some attempt was made to give meaning to the displayed flags.

- F00 is a note to say that zero constant have been calculated.
- F01 is set during the high-pass or lowpass mode, otherwise it's clear.
- F02 is set for Chebyshev, clear for Butterworth.
- F03 denotes the bandwidth is specified at -3 dB. If it's clear, then the BW is specified at -edB by the user and can be anything from near zero to -3 dB. This only occurs during the run of LBL "C?" wherein the "?" is a reminder to key in the ripple / bandwidth measurement figure. "C3" is a Chebyshev label under which the bandwidth is spec'd at -3 dB.
- F04 is set if the user failed to specify a ripple figure for either Chebyshev case. Here, line .102 injects 1 dB and the printout shows "DEF"for default. See the #2 CHB Band-stop listing for an example of this.
- F05 is used internally to tell the difference between a bandpass and a band-stop filter. It's set or cleared by the input to "T" -1 and -2 will run a band-stop; 1, 2, etc. will run a bandpass if F 01 is clear.

If at some time the 41 puts you in "N" after pressing "T" it's because no constants have been done. Inputs such as 3, 4 or 5 for which there is no corresponding filter, will return you to a clear display and a BEEP. Likewise -1 or -2 (band-stop) will do nothingif F 01 is set (hi/lowpass mode).

Open book 1 to page 45. Here Bruce has run off a type 6 Butterworth. The 41 version takes frequencies in MHz so enter:

1 EEX 4, ENTER^, EEX 6, /, 500, ENTER^, EEX 6, / , press "FB" [**A**].

"n"= 5 elements, key in 5 and press "N"(fb). Next press BC [**B**] to generate these 5 constants. There is a wait there while this is being done. A BEEP signifies it's ready to continue. The impedance is 50 ohms, key in 50 and press "Z" [**D**]. Finally, key in the type # (6) and press "T" [**E**].

The next example is on page .58. It's a type-11 bandpass.

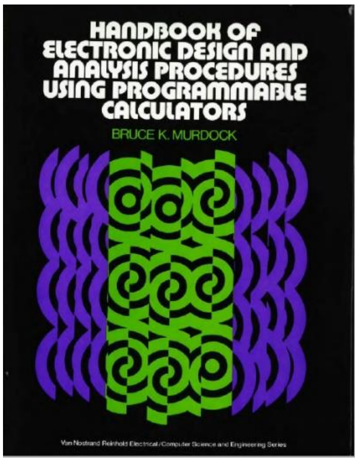
1, ENTER^, 0.02, XEQ "FB" (5 is under the last run on "N" so that doesn't need to be rerun). Ripple 0.5, "C?" (FC) wait, 50 (ohms), "Z", 11, "T" (E). The element Q is ten times the Q figure obtained after running FB.

Book 2, page 68. Here a lowpass filter is to be run off. The procedure differs slightly for the high-pass and lowpass in that the frequency is entered into "X" and "Y". This sets F01. Key in the following for the Butterworth constants:

1183.301, ENTER^, EEX 6, /, ENTER^, XEQ "FB"
4, XEQ "N", XEQ "BC" [**B**]
wait.
1000, XEQ "Z", 1, XEQ "T". A high-pass would be .1 T.

This 1 or .1 uses the decimal point as mental note of how the pass band relates to the decimal point. "1." Is a lowpass but ".1" is a high-pass. It's the same for ".2" and "2." I also ran off a type-2 high-pass by keying in ".2" and pressing XEQ "T". If all the information is already in the 41, there's no need to go back and redo "FB" or "N", etc.

On page 76 Bruce's run off both types of band-stop filters:



63, ENTER^, 6, XEQ 'FB",
5, XEQ "N", 1, XEQ "C?"
wait.
75, XEQ "Z"
-1, XEQ "T" (or -2 for a type-2)

Registers:

R00 - f0
R01 - BW
R02 - Q
R03 - n
R04 - ripple
R05 - impedance
R06 – R14: multiple use scratch
R15 – UP: store the Butterworth and Chebyshev constants.

If you decide to modify the program to conform to Bruce's book on designs using programmable calculators, you can move the constants up from R15 making space available for whatever needs arise. Here again some changes are needed: LINE .186, .195, .556. That should about cover it.

Program listing:

```
*LBL "FB"
CF 01
FIX 1
X=Y?
SF 01
STO 01
X<>Y
STO 00
PI
*
2 E6
*
STO 07
RCL 00
RCL 01
/
STO 02
RTN
19*LBL "O"
E1
/
10^X
2
*
E
-
LN
X<>Y
ENTER^
X^2
RCL 00
X^2
-
X<>Y
/
RCL 01
/
ABS
LN
ABS
/
E
+
2
/
*LBL "N"
SF 00
CF 21
FIX 0
RND
STO 03
"N="
ARCL 03
TONE IND 03
AVIEW
PSE
CLX
FIX 2
VIEW L
SF 21
RTN
*LBL "BC"
CF 00
CF 02
SF 03
CF 04
*LBL 13
PI
RCL 03
/
STO 08
E
STO 11
STO 12
STO 13
2
STO 06
FS? 02
RTN
*LBL 14
RCL 13
RCL 08
2
/
*
SIN
2
*
STO 10
XEQ 17
X<=Y?
GTO 14
GTO 03
*LBL "C?"
CF 00
SF 02
CF 03
SF 04
X#0?
CF 04
X=0?
E
STO 04
E1
/
10^X
FRC
SQRT
1/X
ENTER^
X^2
E
-
SQRT
+
LN
RCL 03
/
E^X
ENTER^
1/X
+
2
/
STO 09
XEQ 13
RCL 04
40
/
10^X
X^2
STO Y
E
ST+ Z
-
/
LN
RCL 03
2
*
/
E^X
ENTER^
1/X
-
2
/
STO 10
X^2
STO 14
*LBL 15
RCL 11
RCL 08
2
/
RCL 13
*
SIN
STO 11
*
RCL 06
RCL 10
*
```

```
164 /
165 4
166 *
167 STO 10
168 RCL 14
169 RCL 08
170 RCL 12
171 *
172 SIN
173 X^2
174 +
175 STO 06
176 XEQ 17
177 X<=Y?
178 GTO 15
179 FS?C 10
180 RTN
181 GTO 03
182 *LBL "C3"
183 SF 10
184 XROM "C?"
185 SF 03
186 23.014
187 RCL 09
188 *LBL 16
189 ST* IND Y
190 DSE Y
191 GTO 16
192 GTO 03
193 *LBL 17
194 RCL 12
195 14
196 +
197 RCL 10
198 STO IND Y
199 E
200 ST+ 12
201 2
202 ST+ 13
203 RCL 03
204 *
205 RCL 13
206 RTN
207 *LBL "Z"
208 STO 05
209 RTN
210 *LBL "T"
211 X=0?
212 GTO 03
213 RCL 03
214 FS? 00
215 GTO "N"
216 RDN
217 CF 05
218 X<0?
219 SF 05
220 FS? 05
221 FC? 01
222 GTO 00
223 GTO 03
224 *LBL 00
225 ABS
226 5
227 X=Y?
228 GTO 03
229 DSE X
230 X=Y?
231 GTO 03
232 DSE X
233 X=Y?
234 GTO 03
235 RDN
236 FIX 0
237 CF 09
238 FRC
239 X#0?
240 SF 09
241 X#0?
242 10^X
243 FC? 09
244 LASTX
245 RND
246 SF 12
247 "# "
248 ARCL X
249 FC? 02
250 >" BTW"
251 FS? 02
252 >" CHB"
253 AVIEW
254 FS? 09
255 "HI-"
256 FC? 09
257 "LO-"
258 FC? 01
259 "BAND"
260 FC? 05
261 >"PASS"
262 FS? 05
263 >"STOP"
264 AVIEW
265 CLA
266 ARCL 03
267 >" ELEMENT"
268 AVIEW
269 CLA
270 ARCL 05
271 >" "
272 AVIEW
273 CF 12
274 ENG 3
275 ADV
276 "FREQUENCY: "
277 ARCL 00
278 >" MHz"
279 FS? 01
280 FS? 09
281 PRA
282 "BANDWIDTH: "
283 ARCL 01
284 >" MHz"
285 FC? 09
286 PRA
287 FIX 0
288 RCL 02
289 E1
290 *
291 "ELEMENT Q: "
292 ARCL X
293 FC? 01
294 PRA
295 FC? 02
296 GTO 00
297 FIX 2
298 "BP RIPPLE: "
299 ARCL 04
300 >" dB"
301 FS? 04
302 >""DEF""
303 PRA
304 *LBL 00
305 CLX
306 STO 12
307 STO 13
308 "LC VALUES:-"
309 FC? 03
310 ARCL 04
311 FS? 03
312 >"3"
313 >" dB BW"
314 PRA
315 ENG 3
316 ADV
317 CF 06
318 CF 07
319 23
320 RCL Z
321 FS? 01
322 +
323 FS? 05
324 -
325 SF 12
326 XEQ IND X
327 FS? 01
328 ADV
329 "----"
330 ACA
331 ACA
332 ACA
333 ADV
```

```
334 ADV
335 ADV
336 ADV
337 ADV
338 *LBL 03
339 CLX
340 CLD
341 FIX 2
342 BEEP
343 RTN
344 *LBL 19
345 *
346 STO 13
347 FS? 06
348 VIEW 13
349 RCL 07
350 X^2
351 *
352 1/X
353 VIEW X
354 FC? 06
355 VIEW 13
356 ADV
357 RTN
358 *LBL 20
359 RCL 05
360 X^2
361 ST/ 10
362 ST* 11
363 RTN
364 *LBL 21
365 SF 07
366 *LBL 22
367 RCL 05
368 RCL 02
369 /
370 RCL 07
371 /
372 STO 10
373 RCL 05
374 X^2
375 /
376 STO 11
377 *LBL 23
378 XEQ 32
379 X<=Y?
380 RTN
381 CF 06
382 RCL IND 08
383 FC? 07
384 SF 06
385 FC? 07
386 RCL 11
387 FS? 07
388 RCL 10
389 XEQ 19
390 XEQ 32
391 X<=Y?
392 RTN
393 SF 06
394 RCL IND 08
395 FS? 07
396 RCL 11
397 FC? 07
398 RCL 10
399 FC? 07
400 CF 06
401 XEQ 19
402 GTO 23
403 *LBL 26
404 RCL 05
405 RCL 07
406 /
407 STO 10
408 RCL 05
409 X^2
410 /
411 STO 11
412 RTN
413 *LBL 24
414 XEQ 26
415 FS? 09
416 XEQ 20
417 GTO 27
418 *LBL 25
419 XEQ 26
420 FC? 09
421 XEQ 20
422 *LBL 27
423 XEQ 32
424 X<=Y?
425 RTN
426 RCL IND 08
427 FS? 09
428 1/X
429 RCL 11
430 *
431 VIEW X
432 XEQ 32
433 X<=Y?
434 RTN
435 RCL IND 08
436 FS? 09
437 1/X
438 RCL 10
439 *
440 VIEW X
441 GTO 27
442 *LBL 01
443 SF 07
444 *LBL 02
445 RCL 02
446 RCL 05
447 /
448 RCL 07
449 /
450 STO 10
451 RCL 05
452 X^2
453 *
454 STO 11
455 *LBL 28
456 XEQ 32
457 X<=Y?
458 RTN
459 SF 06
460 RCL IND 08
461 FS? 07
462 RCL 10
463 FC? 07
464 RCL 11
465 FC? 07
466 CF 06
467 XEQ 19
468 XEQ 32
469 X<=Y?
470 RTN
471 CF 06
472 RCL IND 08
473 FS? 07
474 RCL 11
475 FC? 07
476 RCL 10
477 FC? 07
478 SF 06
479 XEQ 19
480 GTO 28
481 *LBL 07
482 SF 07
483 *LBL 06
484 XEQ 32
485 RCL IND 08
486 STO 14
487 RCL 05
488 /
489 RCL 07
490 /
491 STO 10
492 1/X
493 RCL 07
494 X^2
495 /
496 RCL 02
497 /
498 STO 11
499 FS? 07
500 GTO 29
501 RCL 05
502 X^2
503 ST* 10
504 ST/ 11
```

```
*LBL 29
XEQ 32
X<=Y?
GTO 12
XEQ 30
GTO 29
*LBL 12
SF 06
XEQ 30
RTN
*LBL 30
FC? 07
VIEW 11
FS? 06
GTO 12
RCL 14
RCL IND 08
STO 14
*
1/X
SQRT
STO 09
*LBL 12
FS? 06
0
RCL 02
X<>Y
-
RCL 13
-
RCL 10
*
VIEW X
FS? 07
VIEW 11
ADV
FS? 06
RTN
RCL 09
RCL 10
*
*LBL 31
VIEW X
ADV
RCL 09
STO 13
RTN
*LBL 32
E
ST+ 12
RCL 12
14
+
STO 08
RCL 12
RCL 03
E
+
RTN
*LBL 10
SF 05
*LBL 08
XEQ 34
GTO 33
*LBL 11
SF 05
*LBL 09
SF 07
XEQ 34
RCL 05
X^2
ST* 06
ST* 10
ST/ 11
*LBL 33
XEQ 32
X<=Y?
GTO 12
XEQ 36
GTO 33
*LBL 12
SF 06
XEQ 32
XEQ 36
RTN
*LBL 34
XEQ 32
RCL IND 08
STO 14
RCL 05
*
STO 11
RCL 07
ST/ 11
*
1/X
STO 10
RCL 02
/
STO 06
SF 08
RTN
*LBL 35
FS? 08
GTO 12
RCL 09
CHS
RCL 11
*
GTO 31
*LBL 12
RCL 10
RCL 09
/
GTO 31
*LBL 36
FC? 07
VIEW 06
FS? 06
GTO 12
RCL 14
RCL IND 08
STO 14
*
SQRT
1/X
FC? 05
GTO 12
FC?C 08
SF 08
FC? 08
CHS
*LBL 12
STO 09
RCL 02
FC? 06
+
RCL 13
+
RCL 11
*
VIEW X
FS? 07
VIEW 06
ADV
FS? 06
RTN
FS? 05
GTO 35
RCL 10
RCL 09
/
GTO 31
END
```

PRISMA Filters.

Noise Factor: -- "Best of Prisma"

<u>Rauschoptimierung</u>

RF SIZE 023 USER
445 Bytes

00 = I_c = Kollektorstrom in mA
01 = B = Wechselspannungsverstärkungsfaktor ß
02 = R_{bb} = Basisbahnwiderstand in Ω
03 = R_e = äußere Emitterimpedanz in Ω
04 = R_G = Generatorwiderstand in Ω

Taste A optimiert auf minimales Rauschen, Taste B errechnet faktor bei vorgegenem Kollektorstrom

$$Ø6 = 25,9 \frac{Ø2}{ØØ}$$

$$Ø7 = \frac{25,9}{ØØ} + (Ø3)$$

$$Ø8 = Ø2 + Ø4$$

$$Ø9 = \frac{Ø8 + Ø6}{Ø6}$$

$$U_R = \frac{Ø9^2}{Ø4} \left(Ø7 + \frac{Ø8}{Ø9}\right)$$

$$RF = 20x\lg (U_R)$$

```
01*LBL "NOIZ"
02 ADV
03 SIZE?
04 14
05 X>Y?
06 PSIZE
07 XEQ 11
08 SF 12
09 "NOISE FACTOR"
10 AVIEW
11 CF 12
12 XEQ 11
13*LBL B
14 "IC=?"
15 PROMPT
16 STO 00
17 SF 00
18*LBL A
19 FIX 0
20 RCL 01
21 "B"
22 XEQ 10
23 STO 01
24 RCL 02
25 "Rbb"
26 XEQ 10
27 STO 02
28 RCL 03
29 "Re"
30 XEQ 10
31 STO 03
32 RCL 04
33 "RG"
34 XEQ 10
35 STO 04
36 RCL 10
37 "b="
38 ARCL X
39 >"? KHZ"
40 PROMPT
41 STO 10
42 RCL 11
43 FIX 1
44 "Ue="
45 ARCL X
46 >"? MV"
47 PROMPT
48 STO 11
49 25.9
50 RCL 01
51 *
52 STO 05
53 RCL 02
54 RCL 04
55 +
56 STO 08
57 CLX
58 FC? 00
59 STO 00
60 STO 16
61 13
62 STO 15
63 .1
64 STO 14
65 FIX 2
66 FC? 00
67 GTO 01
68*LBL 00
69 RCL 05
70 RCL 00
71 CF 21
72 VIEW X
73 /
74 STO 06
75 RCL 01
76 /
77 RCL 03
78 +
79 STO 07
80 RCL 08
```

```
81 ENTER^
82 ENTER^
83 RCL 06
84 +
85 LASTX
86 /
87 STO 09
88 /
89 RCL 07
90 +
91 RCL 09
92 X^2
93 RCL 04
94 /
95 *
96 FS? 00
97 X<>Y
98 FC?C 00
99 RTN
100 GTO 04
101*LBL 03
102 STO 16
103 ST+ 00
104 XEQ 00
105 ENTER^
106 X<> 15
107 X<>Y
108 RTN
109*LBL 01
110 RCL 14
111 XEQ 03
112 X<Y?
113 GTO 01
114 10
115 CHS
116 ST/ 14
117*LBL 02
118 RCL 14
119 XEQ 03
120 X<Y?
121 GTO 02
122*LBL 04
123 X<>Y
124 LOG
125 10
126 *
127 STO 12
128 CF 00
129 SF 21
130 TONE 9
131 FC? 55
132 GTO 05
133 FIX 0
134 ADV
135 "B="
136 ARCL 01
137 PRA
138 "Rbb="
139 ARCL 02
140 PRA
141 "Re="
142 ARCL 03
143 PRA
144 "RG="
145 ARCL 14
146 PRA
147 FIX 2
148 "b="
149 ARCL 10
150 >" KHZ"
151 PRA
152 "Ue="
153 ARCL 11
154 >" MV"
155 PRA
156*LBL 05
157 RCL 12
158 "F: "
159 ARCL X
160 >" dB"
161 AVIEW
162 RCL 00
163 RCL 16
164 -
165 "Ic: "
166 ARCL X
167 >" MA"
168 AVIEW
169 1.606 E-11
170 RCL 04
171 *
172 RCL 10
173 *
174 SQRT
175 XEQ 06
176 RCL 12
177 +
178 "RP: "
179 ARCL X
180 >" dB"
181 AVIEW
182 CHS
183 RCL 11
184 XEQ 06
185 +
186 "RA: "
187 ARCL X
188 >" dB"
189 AVIEW
190 XEQ 11
191 GTO A
192*LBL 06
193 775
194 /
195 LOG
196 29
197 *
198 RTN
199*LBL 10
200 "`="
201 ARCL X
202 "`?"
203 PROMPT
204 RTN
205*LBL 11
206 FC? 55
207 RTN
208 "------"
209 TIME
210 TIME
211 TIME
212 TIME
213 ADV
214 END
```

Active Low- and High-Pass Filter, 2nd order

Heinrich Henze – "Best of Prisma"

- The Low Pass Filter program calculates an Active Filter 2nd order. For filters of higher order, the calculation must be carried out for each partial filter with the corresponding coefficients.

- The program Low Pass Filter 2nd order with single coupling calculates the resistances R1, R2 and the capacitor C2,

For high pass filter: XEQ "A2HP"
For low pass filter: XEQ "A2LP"

Operate program on demand.

F = cut-off frequency Hz
A1 = filter-dependent
B1 = filter=dependent
C1 = capacitor values F
C2 = capacitor values F

Outputs:

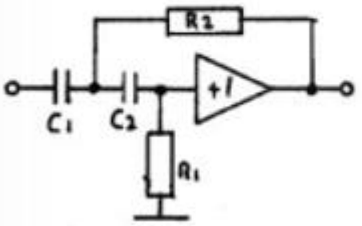
High pass filter: R1, R2
Low pass filter: R1, R2, C2

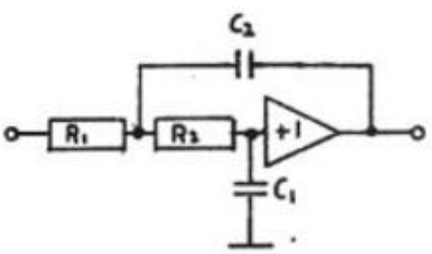
Examples: XEQ "A2HP"

```
F?1.000E3
A1?1.362E0
B1?618.0E-3

C1?10.00E-9
C2?10.00E-9

R1=23.38E3
R2=17.53E3

C1?5.700E-9
C2?10.00E-9

R1=32.19E3
R2=22.34E3
```

XEQ "A2LP"

```
F?1.000E3
A1?1.362E0
B1?618.0E-3

C1?10.00E-9
C2?10.00E-9

R1=7.885E3
R2=13.79E3
C2=14.40E-9

C1?1.000E-9
C2?1.500E-9

R1=72.22E3
R2=144.5E3
C2=1.500E-9
```





Program listing:

```
01*LBL "A2HP"
02 SF 01
03 GTO 00
04*LBL "A2LP"
05 CF 01
06*LBL 00
07 ADV
08 ENG 3
09 "F=?"
10 PROMPT
11 XEQ 01
12 STO 01
13 "A1=?"
14 PROMPT
15 XEQ 01
16 STO 04
17 "B1=?"
18 PROMPT
19 XEQ 01
20 STO 05
21*LBL A
22 ADV
23 "C1=?"
24 PROMPT
25 XEQ 01
26 STO 02
27 "C2=?"
28 PROMPT
29 XEQ 01
30 STO 03
31 ADV
32*LBL B
33 FS? 01
34 GTO 03
35 RCL 04
36 X^2
37 RCL 03
38 X^2
39 *
40 4
41 RCL 05
42 *
43 RCL 02
44 *
45 RCL 03
46 *
47 -
48 X<0?
49 GTO 02
50 SQRT
51 STO 06
52 RCL 04
53 RCL 03
54 *
55 STO 00
56 X<>Y
57 -
58 4
59 PI
60 *
61 RCL 01
62 *
63 RCL 02
64 *
65 RCL 03
66 *
67 STO Z
68 /
69 "R1="
70 XEQ 01
71 RCL 06
72 RCL 00
73 +
74 RCL Z
75 /
76 "R2="
77 XEQ 01
78 RCL 03
79 "C2="
80 GTO 01
81*LBL 02
82 1.2
83 ST* 03
84 GTO B
85*LBL 03
86 2
87 PI
88 *
89 RCL 01
90 *
91 STO 06
92 RCL 04
93 *
94 1/X
95 RCL 02
96 1/X
97 RCL 03
98 1/X
99 +
100 *
101 "R1="
102 XEQ 01
103 RCL 04
104 RCL 05
105 RCL 06
106 *
107 /
108 RCL 02
109 RCL 03
110 +
111 1/X
112 *
113 "R2="
114*LBL 01
115 ARCL X
116 AVIEW
117 FC? 55
118 STOP
119 END
```

Digital Filters Approximation
Thomas Wegmann; Prisma 1985 V3, p31

For some years now, the methods of digital signal processing have been penetrating all areas of electrical engineering. They replace the conventional analog technology in communications, control and power engineering and allow the introduction of systems that cannot be realized in analog. The design of suitable filters plays a major role in this process. Here a program is presented which calculates the most important mathematical parameters for the approximation of digital filters for regulations in the frequency range, whereby a limitation to low-pass systems was made.

The digital system theory knows two different classes of systems: recursive (IIR-) filters with infinitely long impulse response as well as non-recursive (FIR-) systems with an ex-tact definable length of the impulse response, for which there is no analog equivalent.

A brief outline of the filter theory is intended to explain the calculated parameters and explain the program. Figure 1 shows the logical flow diagram. The user is prompted by the program to enter the necessary parameters. These are the pass, stop and sampling frequency (f_S, f_D, f_A). Several branches are possible in the following. If these are to be executed, reply with "**J**", R/S, otherwise the execution is continued with R/S.

With the first branch "FIR?" the system type is specified. By pressing R/S, "IIR" appears in the display, i.e. recursive filter sizes are determined. After entering the tolerance tube parameters 80 and 88 (see Figure 2), the following calculation steps are carried out:

1. Frequency normalization to $f_A/2\pi$, the quantities f_D, fs equivalent to Ω_D, Ω_S in V_D and V_S are displayed as multiples of π.
2. Bilinear transformation transforms Ω_D, Ω_S in V_D and $V_S$and a normalized low-pass filter is introduced: $V_D^* = 1$, $V_S^* = V_S/V_D$.
3. A transformation of the tolerance hue provides an approximation scheme according to Fig. 3, which allows a more pleasant mathematical treatment than Fig. 2. formula (1) applies to the new quantities Delta 1 and Delta 2.

The scheme can be filled out either by functions with a power curve or Chebyshev reaction in the passband or stopband. The required filtering degree NMIN is then determined from equation (2) or (3). For comparison purposes, the filtering degree of an FIR version can also be calculated, which requiredParameter conversion takes place via equation (4). In contrast to IIR systems, non-recursive filters (tolerance diagram Fig. 4) allow targeted progress to be made on the phase response. The simplest solution is to develop a Fourier series, but this is not satisfactory for selective systems due to Gibbs' phenomenon. A windowing of the Fourier coefficients according to KAISER provides in this case a rather simple suboptimal solution.

The degree of the causal filter according to formula (9) is determined from the parameters of the tolerance tube, for which purpose the auxiliary variables are calculated and displayed according to equation (6) to (8). In addition, the parameter ALPHA (formula 1 0) is calculated, which is needed to determine the Kaiser window coefficients from modified Bessel functions. After intermediate calculation according to equation (5) a comparison with IIR systems can be made. Further theory can be found in the extensive literature. The program reacts to the following input errors:

- If fs is not greater than f0, the input is repeated correctly to exclude nonsensical results (negative filter degrees).
- If the respective tolerance tube parameter in the stopband is greater than that in the passband, a correction option is offered, since the stopband is usually subject to stricter tolerance requirements than the passband.
- If the attenuation A of FIR filters is less than 21 dB, the parameter ALPHA is not calculated.

Calculation example (Fix4)

A digital filter with f0=6 kHz, fs=6.77 kHz 80=0.02.88=0.001 is to be sampled with 36 kHz; the characteristics for IIR and FIR realization are sought.

Eingabe	Anzeige
XEQ APPROX	FD=?
6 R/S	FSP?
6,77 R/S	FA=?
36 R/S	FIR?
R/S	IIR
R/S	dD=?
0,02 R/S	dS=?
0,001 R/S	OMEGA D=0,3333 PI
R/S	OMEGA S=0,3761 PI
R/S	VD=0,5774
R/S	VS=0,6707
R/S	VS*=1
R/S	VS*=1,1617
R/S	DELTA 1=0,2031
R/S	DELTA 2=999,9995
R/S	POTTP?
J R/S	N MIN=57
R/S	CHEBY?
J R/S	N MIN=17
R/S	FIR?
J R/S	d1=0,0101
R/S	d2=0,0010
R/S	dOMEGA=0,0428 PI
R/S	A=59,9127
R/S	D=3,6186
R/S	GRAD=170
R/S	ALPHA=5,6436
R/S	IIR?
R/S	ENDE

```
                XEQ "APPROX"
FD=?
                6,0000   RUN
FS=?
                6,7700   RUN
FA=?
               36,0000   RUN
FIR?
                         RUN
IIR
                         RUN
dD=?
                 .0200   RUN
dS=?
                 .0010   RUN
OMEGA D=0,3333 PI
OMEGA S=0,3761 PI
VD=0,5774
VS=0,6707
VD*=1
                         RUN
VS*=1,1617
DELTA 1=0,2031
DELTA 2=999,9995
POTTP?
J                        RUN
N MIN=57
CHEBY ?
J                        RUN
N MIN=17
FIR?
J                        RUN
d1=0,0101
d2=0,0010
dOMEGA=0,0428 PI
A=59,9127
D=3,6186
GRAD=170
ALPHA=5,6436
IIR?
                         RUN
ENDE
```

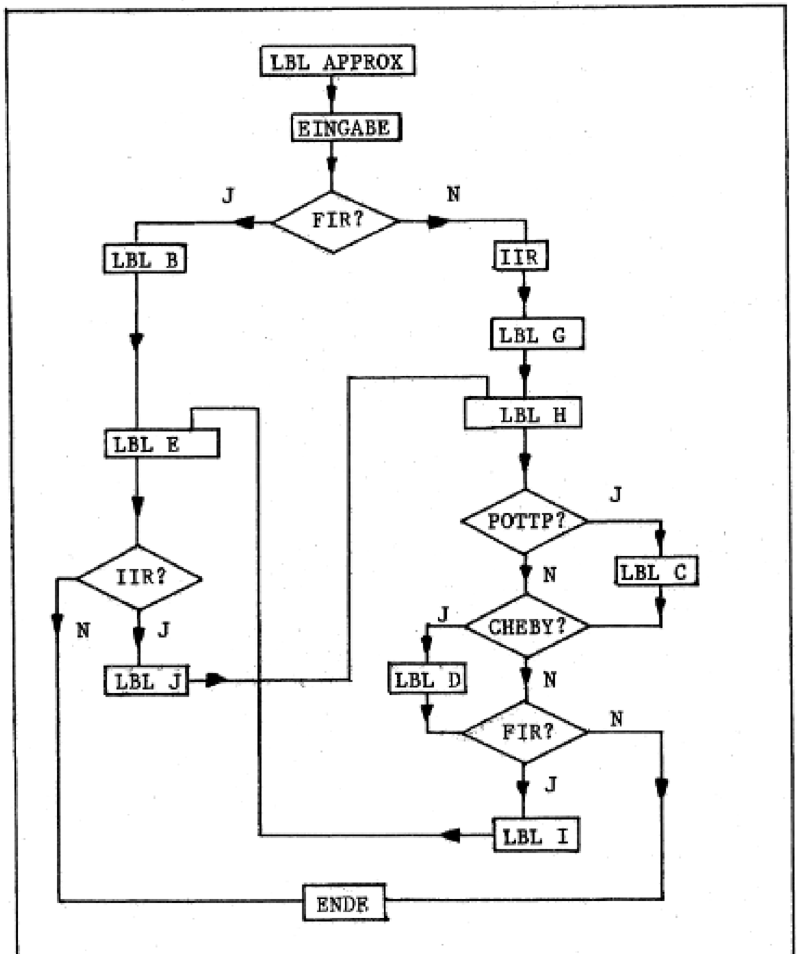


Bild 1: Flußdiagramm

Verwendete Formeln:

$$\Delta 1 = \frac{\sqrt{2\delta_D - \delta_D^2}}{1 - \delta_D} \quad (1a) \qquad \Delta 2 = \frac{\sqrt{1 - \delta_S^2}}{\delta_S} \quad (1b)$$

Potenz – TP: $$N_{MIN} = \left[\text{INT} \frac{\log(\Delta 2/\Delta 1)}{\log v_S^*} \right] + 1 \quad (2)$$

Chebyshev – TP: $$N_{MIN} = \left[\text{INT} \frac{\text{arcosh}(\Delta 2/\Delta 1)}{\text{arcosh } v_S^*} \right] + 1 \quad (3)$$

$$\delta_1 = \frac{\delta_D}{2 - \delta_D} \qquad \delta_2 = \frac{2\delta_S}{2 - \delta_D} \quad (4)$$

$$\delta_D = \frac{2\delta_1}{1 + \delta_1} \qquad \delta_S = 0{,}5(2 - \delta_D)\delta_S \quad (5)$$

$$\Delta\Omega = \Omega_S - \Omega_D \quad (6)$$

$$A = -20 \log\left[\min(\delta_1, \delta_2)\right] \quad (7)$$

$$D = \frac{A - 7{,}95}{14{,}36} \quad (8)$$

$$GRAD = 2\left[\left(INT \frac{\pi D}{\Delta\Omega}\right) + 1\right] \quad (9)$$

$$\alpha = \begin{cases} 0{,}1102(A - 8{,}7) & \text{für } A > 50 \\ 0{,}5842(A - 21)^{0{,}4} + 0{,}07886(A - 21) & \text{für } 21 \leq A \leq 50 \end{cases} \quad (10)$$

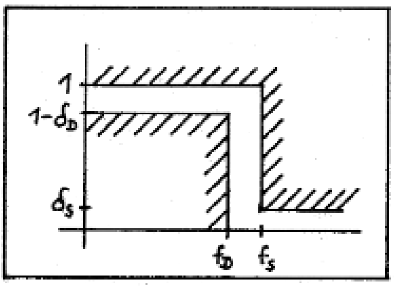


Bild 2:
Toleranzschlauch IIR

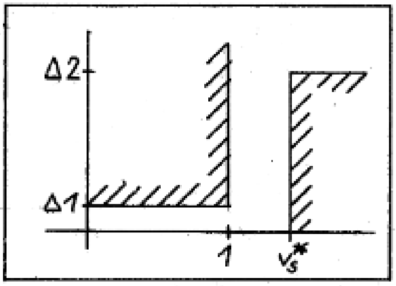


Bild 3:
Modifikation von Bild 2

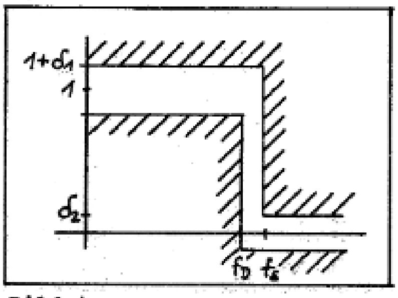


Bild 4:
Toleranzschema FIR

Program listing:

```
01*LBL 06
02 "ERROR: FS<FD"
03 PROMPT
04*LBL "APPROX"
05 SIZE?
06 8
07 X>Y?
08 PSIZE
09 SF 21
10 CF 29
11 "FD=?"
12 PROMPT
13 STO 00
14 "FS=?"
15 PROMPT
16 STO 01
17 X<=Y?
18 GTO 06
19 "FA=?"
20 PROMPT
21 STO 02
22 "FIR"
23 XEQ 02
24 X=Y?
25 GTO B
26 "* IIR *"
27 AVIEW
28*LBL 07
29 "dD=?"
30 PROMPT
31 STO 03
32 "dS=?"
33 PROMPT
34 STO 04
35 X<=Y?
36 GTO G
```

```
37 "dS>dD"
38 XEQ 01
39 X#Y?
40 GTO 07
41*LBL G
42 RCL 00
43 RCL 02
44 /
45 ST+ X
46 "OMG-D"
47 XEQ 10
48 PI
49 *
50 STO 05
51 RCL 01
52 RCL 02
53 /
54 ST+ X
55 "OMG-S"
56 XEQ 10
57 PI
58 *
59 STO 06
60*LBL H
61 RCL 05
62 FS?C 00
63 RTN
64 2
65 /
66 R-D
67 TAN
68 "VD"
69 XROM "="
70 RCL 06
71 2
72 /
73 R-D
74 TAN
75 "VS"
76 XROM "="
77 X<>Y
78 /
79 "VD*=1"
80 PROMPT
81 "VS*"
82 XROM "="
83 STO 07
84 RCL 03
85 ST+ X
86 RCL 03
87 X^2
88 -
89 SQRT
90 E
91 RCL 03
92 -
93 /
94 STO 00
95 "<>1"
96 XROM "="
97 E
98 RCL 04
99 X^2
100 -
101 SQRT
102 RCL 04
103 /
104 STO 01
105 "<>2"
106 XROM "="
107 "POTTP"
108 XEQ 02
109 X=Y?
110 XEQ C
111 "CHEBY"
112 XEQ 02
113 X=Y?
114 XEQ D
115 "FIR"
116 XEQ 02
117 X#Y?
118 GTO 00
119 GTO I
120*LBL C
121 RCL 01
122 RCL 00
123 /
124 LOG
125 RCL 07
126 LOG
127 GTO 05
128*LBL D
129 RCL 01
130 RCL 00
131 /
132 ACOSH
133 RCL 07
134 ACOSH
135*LBL 05
136 /
137 INT
138 E
139 +
140 FIX 0
141 "N.MIN"
142 XROM "="
143 FIX 4
144 RTN
145*LBL B
146 "d1=?"
147 PROMPT
148 STO 03
149 "d2=?"
150 PROMPT
151 STO 04
152 X<=Y?
153 GTO A
154 "d2>d1"
155 XEQ 01
156 X#Y?
157 GTO B
158*LBL A
159 SF 00
160 XEQ G
161*LBL E
162 -
163 PI
164 /
165 "dOMG"
166 XEQ 10
167 STO 07
168 RCL 03
169 RCL 04
170 X>Y?
171 RCL Y
172 LOG
173 -20
174 *
175 STO 02
176 "A"
177 XROM "="
178 20
179 X<=Y?
180 SF 00
181 RCL 02
182 7.95
183 -
184 14.36
185 /
186 "D"
```

```
187 XROM "="
188 RCL 07
189 /
190 INT
191  E
192 +
193 FIX 0
194 ST+ X
195 "GRAD"
196 XROM "="
197 FIX 4
198 50
199 RCL 02
200 X<=Y?
201 GTO 03
202 8.7
203 -
204 .1102
205 *
206 GTO 04
207*LBL 03
208 FS?C 00
209 GTO 00
210 21
211 -
212 STO \
213 .4
214 Y^X
215 .5842
216 *
217 RCL \
218 .07886
219 *
220 +
221*LBL 04
222 "ALPHA"
223 XROM "="
224 "IIR"
225 XEQ 02
226 X#Y?
227 GTO 00
228 GTO J
229*LBL I
230 RCL 03
231 ENTER^
232 CHS
233 2
234 +
235 1/X
236 *
237 STO 03
238 "d1"
239 XROM "="
240 LASTX
241 ST+ X
242 RCL 04
243 *
244 STO 04
245 "d2"
246 XROM "="
247 RCL 06
248 RCL 05
249 GTO E
250*LBL J
251 RCL 03
252 RCL 03
253  E
254 +
255 1/X
256 ST+ X
257 *
258 STO 03
259 "dD"
260 XROM "="
261 CHS
262 2
263 +
264 2
265 /
266 RCL 04
267 *
268 STO 04
269 "dS"
270 XROM "="
271 GTO H
272*LBL 10
273 >"="
274 ARCL X
275 >" PI"
276 AVIEW
277 RTN
278*LBL 01
279 >", OK"
280*LBL 02
281 AON
282 >" ?"
283 PROMPT
284 AOFF
285 ATOX
286 89
287 RTN
288*LBL 00
289 CLST
290 SF 29
291 "DONE"
292 AVIEW
293 END
```

Butterworth High-pass and Low-pass filters:
Werner Meschede, Prisma 1988 V2 p40

This program was created by porting a BASIC program from the book "Nolte-Basic: HF-computing programs" published as volume 9 in the series "Franzis Computer-Bibliothek".The pictures shown are also from this book. For the kindly provided material I'm thankful.

This program calculates the elements of a passive low- or high-pass filter with Butterworth transfer characteristic.

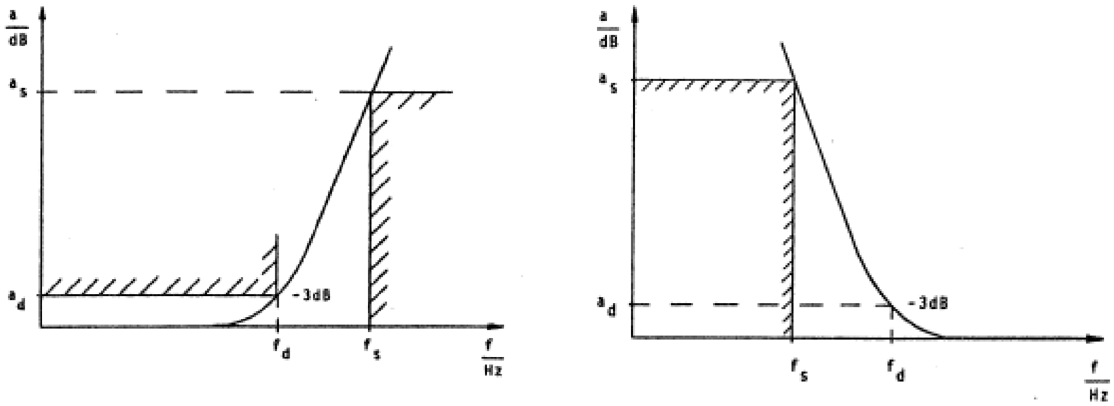


Fig. 1 attenuation curve of Butterworth low-pass (left) anf High-pass (right) filters

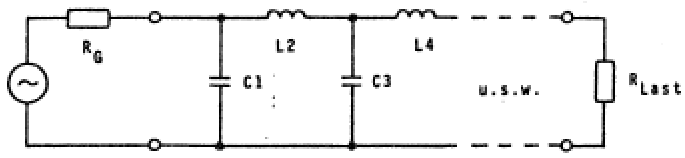


Fig. 2 Schematic diagram of a passive Butterworth low-pass filter

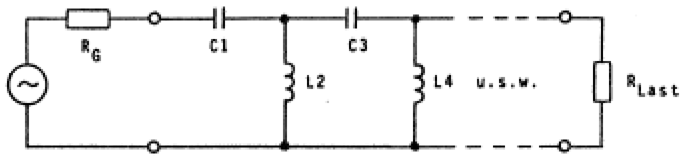


Fig 3 Schematic diagram of a passive Butterworth high-pass filter

The program uses the flags 01 and 29 and the data registers00 to 06 for internal tasks.

Operation:

<XEQ> "BWLP" for Butterworth low-pass filter; or rather
<XEO> "BWHP" for Butterworth high-pass filter

1. Input and output resistance "RO" <R/S> [Ohm]
2. Input of the operating frequency "FO" <R/S> [Hz]
3. Enter the blocking frequency "FS" <R/S> [Hz]
4. Enter the attenuationat blocking frequency "A" <R/S> [dB]
5. The program presents the results: the number of needed elements and their values.

Examples:

```
Beispiel 1 :
      XEQ "BTP"
      R0 =?
       50,0000 RUN
      F0 =?
       145+06 RUN
      FS =?
       430+06 RUN
      A =?
       40,0000 RUN
      N=5
      RUN
      C(1)=13,57E-12 F
      RUN
      L(2)=88,80E-9 H
      RUN
      C(3)=43,90E-12 F
      RUN
      L(4)=88,80E-9 H
      RUN
      C(5)=13,57E-12 F
      RUN
```

```
Beispiel 2 :
      XEQ "BHP"
      R0 =?
       50,0000 RUN
      F0 =?
       430+06 RUN
      FS =?
       145+06 RUN
      A =?
       60,0000 RUN
      N=7
      RUN
      C(1)=16,63E-12 F
      RUN
      L(2)=14,84E-9 H
      RUN
      C(3)=4,108E-12 F
      RUN
      L(4)=9,253E-9 H
      RUN
      C(5)=4,108E-12 F
      RUN
      L(6)=14,84E-9 H
      RUN
      C(7)=16,63E-12 F
      RUN
```

Program listing:

```
01*LBL "BWLP"
02 SF 01
03 GTO 00
04*LBL "BWHP"
05 CF 01
06*LBL 00
07 RAD
08 CF 29
09 "R0=?"
10 PROMPT
11 STO 01
12*LBL 01
13 "F0=?"
14 PROMPT
15 STO 02
16 RCL 01
17 *
18 PI
19 *
20 1/X
21 STO 05
22 "FS=?"
23 PROMPT
24 STO 00
25 "A=?"
26 PROMPT
27 RCL 00
28 RCL 02
29 FC? 01
30 X<>Y
31 X>Y?
32 GTO 01
33 /
34 LOG
35 20
36 *
37 /
38 1
39 STO 06
40 +
41 INT
42 STO 04
43 FIX 0
44 "N"
45 XROM "="
46 -2
47 STO 03
48*LBL 02
49 RCL 05
50 RCL 06
51 .5
52 -
53 RCL 04
54 /
55 PI
56 *
57 SIN
58 FS? 01
59 *
60 FS? 01
61 GTO 00
62 4
63 *
64 /
65*LBL 00
66 RCL 06
67 RCL 03
68 XROM "SHOW"
69 -1
70 ST* 03
71 RCL 01
72 RCL 03
73 Y^X
74 ST* 05
75 1
76 ST+ 06
77 RCL 06
78 RCL 04
79 X<Y?
80 GTO 00
81 GTO 02
82*LBL "SHOW"
83 "L("
84 X<0?
85 "C("
86 FIX 0
87 ARCL Y
88 "`)="
89 ENG 3
90 ARCL Z
91 X<0?
92 >" F"
93 X>0?
94 >" H"
95 PROMPT
96 RTN
97*LBL 00
98*LBL "END"
99 CLST
100 CLA
101 CF 01
102 SF 29
103 FIX 4
104 END
```

Chebyshev High-pass and Low-pass filters:
Werner Meschede, Prisma 1987 V6 p62

This program was created by porting a BASIC program from the book "Nolte Basic: RF Calculation Programs" published as volume 9 in the series "Franzis Computer Library". Also the shown pictures are taken from this book. For the material kindly provided I would like to take this opportunity to thank you.

This program calculates the elements of a passive low- or high-pass filter with Chebyshev transmission characteristics.

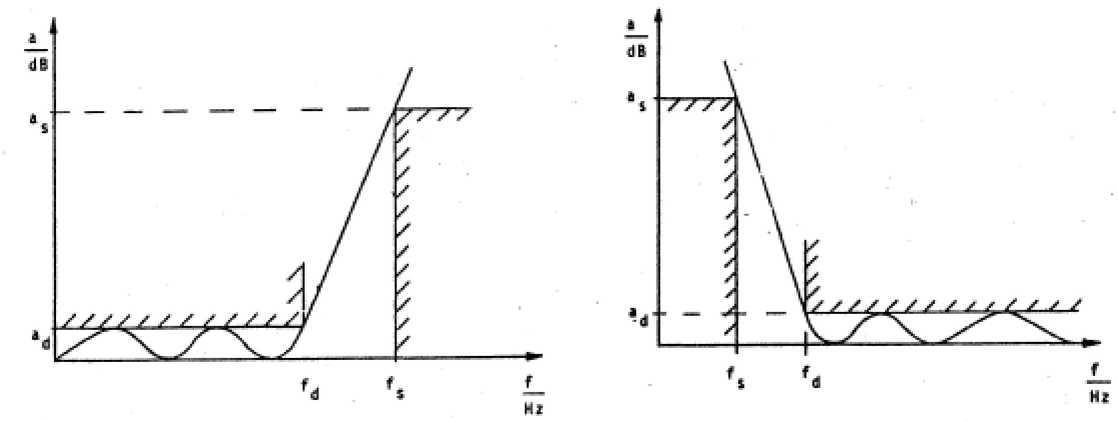


Fig 1. Damping curve of a pole Chebyshev Low-pass (left) and High-pass (right) filters depending on the frequency characteristics.

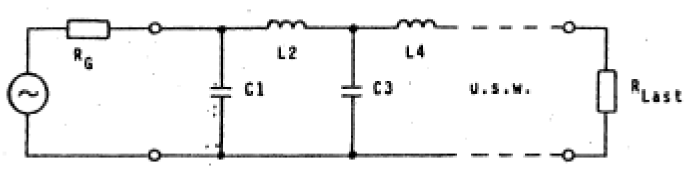


Fig 2. Schematic diagram of a passive low-pass filter

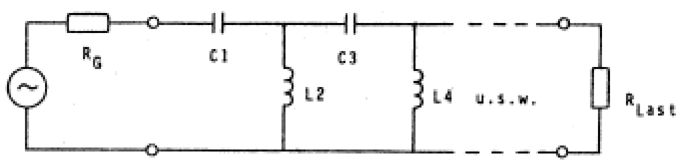


Fig 3. Schematic diagram of a passive high-pass filter

The program uses flag 22 and 29 and the data registers 00 to 10 for internal tasks.

Operation:
<XEQ> "CBLP" for Chebyshev Low-pass filter; or rather
<XEQ> "CBHP" for Chebyshev High-pass filter

1. Input of the input and output resistance, "RO"<R/S> [ohm]
2. Input of the operating frequency"FO"<R/S> [Hz]
3. Input of the return loss"RD"<R/S> [dB]
4. Enter the type of the 1st filter element("L" or "C")
5. "1 ELEMENT" <R/S>
6. Enter the number of elements "N" <R/S>

If the number of elements is to be determined by the program press<R/S> without any input, and then provide the blocking frequency "FS" <R/S> [Hz] and the damping at fs "A (FS)" <R/S> [dB] .

All elements are then output.

```
Beispiel 1 :

XEQ "TTP"
RØ =?
 5Ø,ØØØØ RUN
FØ =?
 3Ø+Ø6 RUN
RD =?
 2Ø,ØØØØ RUN
1. ELEMENT =?
C RUN
N =?
 5,ØØØØ RUN
C(1)=1Ø3,3E-12F
RUN
L(2)=364,ØE-9H
RUN
C(3)=191,3E-12F
RUN
L(4)=364,ØE-9H
RUN
C(5)=1Ø3,3E-12F
RUN
```

```
Beispiel 2 :

XEQ "THP"
RØ =?
 75,ØØØØ RUN
FØ =?
 17Ø+Ø6 RUN
RD =?
 18,ØØØØ RUN
1. ELEMENT =?
L RUN
N =?
RUN
FS =?
 3Ø+Ø6 RUN
A(FS) =?
 8Ø,ØØØØ RUN
N=5
RUN
L(1)=65,99E-9H
RUN
C(2)=9,Ø7ØE-12F
RUN
L(3)=37,Ø8E-9H
RUN
C(4)=9,Ø7ØE-12F
RUN
L(5)=65,99E-9H
RUN
```

Program listing:

```
01*LBL "CBLP"
02  E
03 CHS
04 GTO 00
05*LBL "CBHP"
06  E
07*LBL 00
08 STO 00
09 RAD
10 CF 29
11 "R0=?"
12 PROMPT
13 STO 01
14 "F0=?"
15 PROMPT
16 ST+ X
17 PI
18 *
19 STO 02
20 "RD=?"
21 PROMPT
22 - E1
23 /
24 10^X
25  E
26 -
27 CHS
28 1/X
29 STO 03
30*LBL 02
31 "1ST.? L/C"
32 AON
33 PROMPT
34 AOFF
35 ASTO X
36 "L"
37 ASTO Y
38 X#Y?
39 GTO 00
40  E
41 GTO 01
42*LBL 00
43 "C"
44 ASTO Y
45 X#Y?
46 GTO 02
47 -1
48*LBL 01
49 STO 04
50 CF 22
51 "N=?"
52 PROMPT
53 FS? 22
54 GTO 00
55 "FS=?"
56 PROMPT
57 ST+ X
58 PI
59 *
60 1/X
61 RCL 02
62 *
63 RCL 00
64 Y^X
65 ENTER^
66 X^2
67  E
68 -
69 SQRT
70 +
71 STO 05
72 "A(FS)=?"
73 PROMPT
74  E1
75 /
76 10^X
77  E
78 -
79 4
80 RCL 03
81  E
82 -
83 /
84 *
85 2
86 -
87 STO 07
88 LN
89 RCL 05
90 LN
91 /
92 STO 06
93*LBL 03
94 RCL 05
95 RCL 06
96 Y^X
97 ENTER^
98 ENTER^
99 1/X
100 +
101 RCL 07
102 -
103 RCL Y
104 RCL X
105 1/X
106 -
107 /
108 RCL 05
109 LN
110 1/X
111 *
112 ST- 01
113 ABS
114 .01
115 X<Y?
116 GTO 03
117 RCL 06
118 2
119 /
120 INT
121  E
122 +
123 FIX 0
124 "N"
125 XROM "="
126*LBL 00
127 STO 07
128  E
129 2
130 RCL 03
131 SQRT
132  E
133 +
134 /
135 -
136 1/X
137 RCL 07
138 ST+ X
139 1/X
140 Y^X
141 ENTER^
142 1/X
143 -
144 2
145 /
146 STO 06
```

```
147 PI
148 2
149 /
150 RCL 07
151 /
152 SIN
153 STO 03
154 ST+ X
155 X<>Y
156 /
157 STO 05
158  E
159 STO 08
160*LBL 04
161 RCL 01
162 RCL 04
163 Y^X
164 RCL 02
165 /
166 RCL 05
167 RCL 00
168 Y^X
169 /
170 RCL 08
171 RCL 04
172 XROM "SHOW"
173 -1
174 ST* 04
175 ST- 08
176 RCL 07
177 RCL 08
178 X>Y?
179 GTO "END"
180 RCL 03
181 STO 09
182 RCL 05
183 STO 10
184 R^
185 R^
186 2
187 1/X
188 -
189 X<>Y
190 /
191 PI
192 *
193 SIN
194 STO 03
195 RCL 09
196 *
197 4
198 *
199 RCL 08
200  E
201 -
202 PI
203 *
204 RCL 07
205 /
206 SIN
207 X^2
208 RCL 06
209 X^2
210 +
211 /
212 RCL 10
213 /
214 STO 05
215 GTO 04
216 END
```

Pass-band and Stop-band Filters.

Werner Meschede, Prisma 1988 V1 p44

This program was created by porting a BASIC program from the book " No1le-Basic: HF-calculation program" published as volume 9 in the series "Franzis Computer Library" . Also the shown Pictures are taken from this book. For the material kindly provided I would like to take this opportunity to thank you.

This program breaks the elements of a passive band-pass or band-reject filter with Chebyshev transmission characteristics. Realistic component values up to a ratio of 10:1 of operating frequency to bandwidth. For narrower band applications this circuit is not suitable.

The program uses the flag 01 and 29 and the data registers 00 to 10 for inteme tasks.

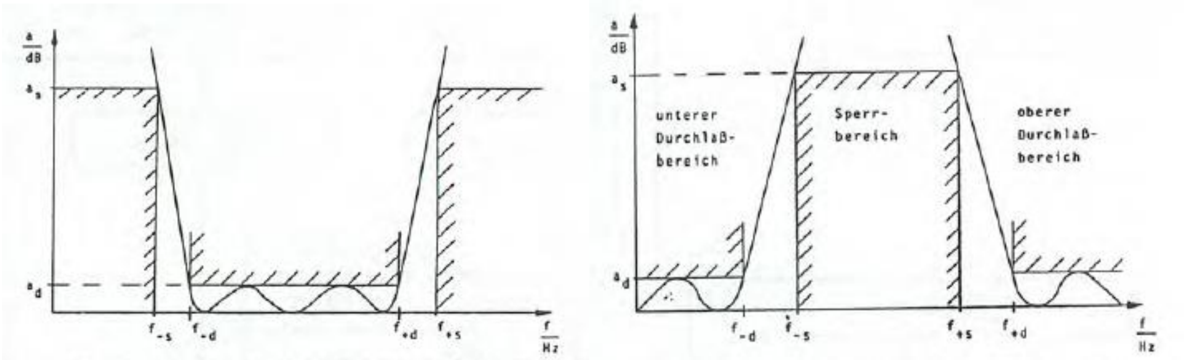


Fig 1. Attenuation curve of a Chebyshev bandpass (left) and band-stop (right) filters as a function of the frequency

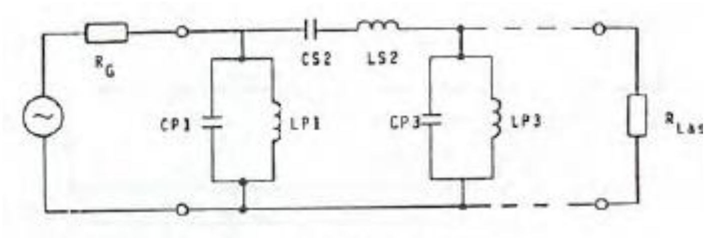


Fig 2. Schematic diagram of a passive bandpass filter

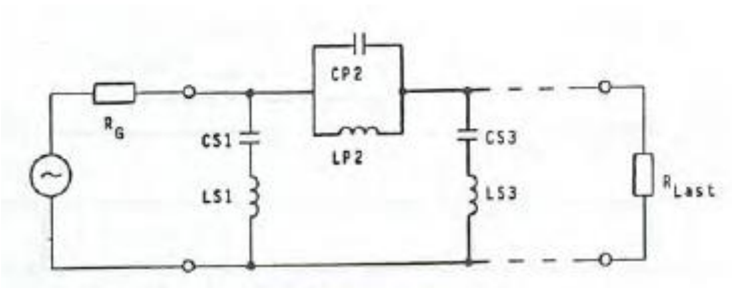


Fig 3. Schematic diagram of a passive band elimination filter

Operation:

< XEQ> "CBPB "for Chebyshev Pass-band, or:
< XEQ> " CBSB" for Chebyshev Stop-Band

1. Input of the input and output resistance "RO"< RIS> [Ohm]:
2. Input of the operating frequency .. FO" < RIS> [Hz]:
3. Input of bandwidth "b.P' <AIS> [Hz] :
4. Input of the return flow attenuation "AD" <RIS> (dB]:
5. Input of number of elements "N"<A/S>.

Then all elements are output

Examples:

```
XEQ "TBP"
R0 =?
 75,0000 RUN
F0 =?
 200+06 RUN
ΔF =?
 60+06 RUN
RD =?
 18,0000 RUN
N =?
 3,0000 RUN
CP(1)=33,49E-12 F
RUN
LP(1)=18,91E-9 H
RUN
CS(2)=2,811E-12 F
RUN
LS(2)=225,3E-9 H
RUN
CP(3)=33,49E-12 F
RUN
LP(3)=18,91E-9 H
RUN
```

```
XEQ "TBS"
R0 =?
 75,0000 RUN
F0 =?
 98+06 RUN
ΔF =?
 20+06 RUN
RD =?
 18,0000 RUN
N =?
 3,0000 RUN
CS(1)=4,184E-12 F
RUN
LS(1)=630,3E-9 H
RUN
CP(2)=93,70E-12 F
RUN
LP(2)=28,15E-9 H
RUN
CS(3)=4,184E-12 F
RUN
LS(3)=630,3E-9 H
RUN
```

01*LBL "CBPB"
02 SF 01
03 -1
04 GTO 00
05*LBL "CBSB"
06 CF 01
07 E
08*LBL 00
09 STO 03
10 RAD
11 CF 29
12 *"R0=?"*
13 PROMPT
14 STO 01
15 *"F0=?"*
16 PROMPT
17 ST+ X
18 PI
19 *
20 X^2
21 STO 02
22 "aF=?"
23 PROMPT
24 ST+ X
25 PI
26 *
27 STO 00
28 *"RD=?"*
29 PROMPT
30 -10
31 /
32 10^X
33 E
34 -
35 CHS
36 1/X
37 SQRT
38 E
39 +
40 1/X
41 ST+ X
42 E
43 -
44 CHS

```
45 1/X
46 STO 04
47 "N=?"
48 PROMPT
49 STO 07
50 ST+ X
51 1/X
52 RCL 04
53 X<>Y
54 Y^X
55 ENTER^
56 1/X
57 -
58 2
59 /
60 STO 04
61 PI
62 2
63 /
64 RCL 07
65 /
66 SIN
67 STO 05
68 X<>Y
69 /
70 ST+ X
71 STO 06
72  E
73 STO 08
74*LBL 04
75 RCL 00
76 RCL 06
77 FS? 01
78 1/X
79 *
80 RCL 02
81 /
82 RCL 01
83 RCL 03
84 Y^X
85 /
86 RCL 01
87 RCL 03
88 Y^X
89 RCL 00
90 /
91 RCL 06
92 FS? 01
93 1/X
94 /
95 RCL 03
96 X<0?
97 "P"
98 X<0?
99 GTO 00
100 "S"
101 RDN
102 X<>Y
103 R^
104*LBL 00
105 XEQ 10
106 -1
107 ST* 03
108 ST- 08
109 RCL 07
110 RCL 08
111 X>Y?
112 GTO "END"
113 RCL 05
114 STO 09
115 RCL 06
116 STO 10
117 R^
118 R^
119 ST+ X
120  E
121 -
122 X<>Y
123 /
124 PI
125 *
126 2
127 /
128 SIN
129 STO 05
130 RCL 09
131 *
132 4
133 *
134 RCL 10
135 /
136 RCL 08
137  E
138 -
139 PI
140 *
141 RCL 07
142 /
143 SIN
144 X^2
145 RCL 04
146 X^2
147 +
148 /
149 STO 06
150 GTO 04
151*LBL 10
152 ASTO X
153 "C"
154 XEQ 09
155 X<>Y
156  E6
157 *
158 X<>Y
159 ARCL Y
160 >"µF"
161 PROMPT
162 "L"
163 XEQ 09
164 RCL Z
165  E3
166 *
167 ARCL X
168 >"MH"
169 PROMPT
170 RTN
171*LBL 09
172 ARCL X
173 >"("
174 FIX 0
175 ARCL 08
176 >")="
177 ENG 3
178 END
```

Network Frequency Response Analysis. [*by Michael Moser*]

From the User's Program Library Europe #10786, included in the EE Filters module.

This program computes the frequency response on a desired interval of a general linear network made up of resistors, capacitors, inductors and voltage-controlled dependent current sources. You define the circuit by keying in the number of nodes (i.e. the order), the types and values of the components, and the nodes they're connected to.

The program consists of three blocks: (1) the input stage, (2) the construction of the admittance matrix, and (3) a matrix reduction stage. Then successive sweeps will loop showing the different electrical outputs with each frequency value.

Let w = 2□f, with f the work frequency. The elements of the admittance matrix are formed as follows for the different components:

Resistors :	Y(R) = 1/R with R in ohms
Capacitors:	Y(C) = jwC, with C in farads
Inductors:	Y(L) = -j/wL, with L in Henries
Voltg-Cntl'd:	Y(VCS) = gm , the transconductance

The following conventions are to be observed:

- *Node "0" is the reference or ground node*; node "1" is the input, and node "2" is the output. A 1-volt reference source is connected between ground and the input node. the phase output is always between +/-180 deg
- For R, L, C components the "from" node cannot be the ground.
- Problems may occur with inductors in networks at near-zero frequencies. For DC analysis you should redesign the network, shorting all inductors and specify the starting frequency to be zero.
- *Independent current sources are not supported*. All voltage-controlled current sources (VCS) are specified so that the voltage is measured between the node and ground, and the current leaves ground and enters the "To:" node.

Once the complex matrix admittance (NMA) is formed, the following matrix equations can be used to describe the steady-state performance of a network at a given frequency:

$[\ Y\ ]_{nxn}\ [\ V\ ]_{nx1} = [\ I\ ]_{nx1}$, and solving for the voltages:

$[\ V\ ]_{nx1} = [\ Y\ ]^{-1}_{nxn}\ [\ I\ ]_{nx1}$

Since for this program is only necessary to find one mode voltage, the complex matrix inversion method is replaced in favor of a faster and simpler matrix-reduction approach. First a partition is made, whereby the NAM is expressed as follows:

	[Y11] | [Y12]	Y11 is a square submatrix (n-1)x(n-1)
$[\ Y\ ]_{nxn}$ =	------- | --------	Y12 is a column vector (n-1)x1
	[Y21] | [Ynn]	Y21 is a row vector for (n-1)-th. element)
		Ynn is the selected element of the NAM

This is followed by a order reduction of $[\ Y\ ]_{nxn}$ to $[\ Y^{(1)}\ ]_{(n-1)x(n-1)}$ as follows:

$$[\ Y^{(1)}\ ]_{(n-1)x(n-1)} = [\ Y11]\ \ - [\ Y12][Y21]\ [Ynn]^{-1}$$

The order reduction step is repeated n-2 times, until we obtain a 2x2 matrix:

$$[\ Y^{(n-2)}\ ]_{2x} = \begin{array}{c|c} [Y'11] & [Y'12] \\ \hline [Y'21] & [Y'nn] \end{array}$$

The required transfer function is then given by the expression: V2/V1 = -[Y'21] / [Y'22]

Which is expressed as a magnitude in dB = 20 log |V2/V1|.

Program details.

The input routines offer a very convenient method to enter and change the component parameters. The modifications made use functions PMTK and ARCLI from the AMC_OS/X module, which therefore needs to be plugged in as well. A few program options include:

- Press **[H]** for a new circuit
- Press [**J**] to change the increment mode from Linear to Logarithmic
- Press [**G**] to change the increment step or the number of points per decade
- Press [**F**] to change the start frequency (Fa) or end frequency (Fe)
- Press [**E**] to change a component value (error corrections or design changes)

The prompts will request to enter the element type first, followed by the "from-to" node configuration and element value. Voltage-controlled sources differ from that scheme in that they require the dependent voltage instead.

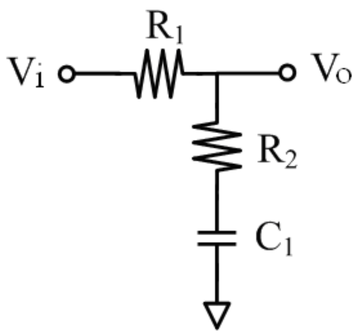
,


--


Example1.

For the passive RC-filter shown below, determine the frequency response from 1MHz to 15MHz at 2Mhz increments. The component parameter values are:

R1 = 50 Ω; C = 1 nF; R2 = 20 Ω



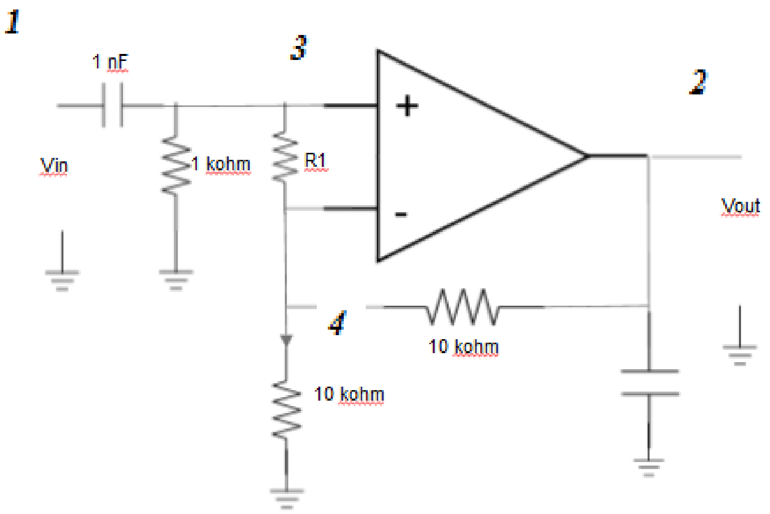
The analytical expression for the transfer function is:Vout / Vin = $[1+ j\ \omega.C_1.R_2]/[1+j\ \omega.C_1(R_1+R_2)]$

The results are shown in the table below:

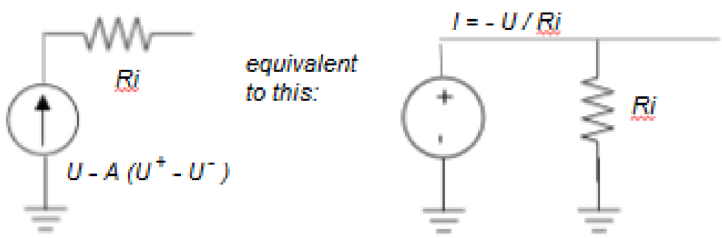
f (Hz)	IGI (dB)	<)G (deg)	f (Hz)	IGI (dB)	<)G (deg)
1 E6	-0.700	-16.5785	9 E6	-8.641	-27.305
3 E6	-3.802	-32.186	11 E6	-9.235	-24.205
5 E6	-6.216	-33.4055	13 E6	-9.630	-21.552
7 E6	-7.714	-30.670	15 E6	-9.903	-19.328

Example2.

Compute the frequency response of the active high-pass filter shown below from 10 Hz to 10 KHz with 3 points per decade. The values of R1 = 10 kΩ; C2 = 50 µF; A= 10 E6



There is a problem concerning the Op-Amp in this circuit. An ideal Op-Amp is defined as a voltage-controlled *voltage source*, and this type of source is not supported by the NAM, node admittance model, which can only handle voltage-controlled *current sources.* But it's possible to transform the Op-Amp into that type adding an inner resistance in parallel, Ri. Therefore the Op-Amp can be replaced with the following circuit, where I = A.(U^+ - U^-) / Ri



Thus the OpAmp is replaced by two current sources (each controlled by the input voltages of the OpAmp), plus the inner resistance - *all connected between ground and node #2*:

$I_1 = - A\,U^{(+)} / Ri$, ;   and: $I_2 = A.U^{(-)} / Ri$

In closed-loop circuits like this one the value of this resistor is not critical; it just has to be small enough to not have noticeable influence. In this example we've chosen an inner resistance of 0.1Ω – which is very small, compared to the other resistors in the circuit. You can also use an inner resistance of 0.01 Ω to check the results, which are practically the same.

The results are shown in the table below:

f (Hz)	IGI (dB)	<)G (deg)	f (Hz)	IGI (dB)	<)G (deg)
10.000	-18.033	86.405	464.16	5.538	18.926
21.544	-11.428	82.291	1.0000 E3	5.912	9.043
46.416	-5.037	73.741	2.1544 E3	5.997	4.225
100.00	0.539	57.858	4.6416 E3	6.016	1.964
215.44	4.129	36.454	10.000 E3	6.020	0.912

Program Listing.-

```
1   *LBL "NETFR"
2   CF 21
3   E
4   STO 04
5   RCL 03
6   "ORDER=?"
7   ASTO 05
8   PROMPT
9   CLA
10  ARCL 05
11  ARCLI
12  AVIEW
13  PSE
14  STO 03
15  X^2
16  ST+ X
17  22
18  +
19  STO 06
20  E
21  +
22  STO 07
23  SIZE?
24  X<>Y
25  X>Y?
26  PSIZE
27  CF 00
28  ADV
29  "INPUT:"
30  AVIEW
31  *LBL 87
32  "TYPE? RLCV"
33  PMTK
34  GTO IND X
35  *LBL "R "
36  *LBL 01
37  SF 03
38  38"R="
39  GTO 00
40  *LBL "L "
41  *LBL 02
42  "L="
43  GTO 00
44  *LBL "C "
45  *LBL 03
46  "C="
47  GTO 00
48  *LBL "VCS"
49  *LBL 04
50  SF 02
51  "<VCS"
52  *LBL 00
53  ASTO 05
54  GTO 86
55  *LBL e
56  ASHF
57  >"=?"
58  PROMPT
59  *LBL E
60  SF 00
61  "CHANGE #."
62  FC?C 22
63  GTO e
64  X<=0?
65  GTO e
66  ARCLI
67  AVIEW
68  PSE
69  X<> 04
70  STO 00
71  RCL 04
72  RCL 03
73  X^2
74  *LBL 10
75  +
76  ST+ X
77  20
78  +
79  STO 06
80  E
81  +
82  STO 07
83  GTO 87
84  *LBL J
85  *LBL 05
86  ADV
87  "SCALE? LG"
88  PMTK
89  GTO IND X
90  *LBL 03
91  FS? 01
92  GTO 01
93  *LBL "LOG"
94  *LBL 02
95  CF 01
96  96"PTS/DK=?"
97  ASTO 05
98  RCL 02
99  X#0?
100 LOG
101 X#0?
102 1/X
103 PROMPT
104 "LOG: "
105 ARCLI
106 >" "
107 ARCL 05
108 AVIEW
109 1/X
110 10^X
111 STO 02
112 GTO F
113 *LBL "LIN"
114 *LBL 01
115 SF 01
116 ENG 3
117 "STEP="
118 ASTO 05
119 >"?"
120 RCL 02
121 PROMPT
122 122"LIN: "
123 ARCL 05
124 ARCLI
125 >" HZ"
126 AVIEW
127 PSE
128 STO 02
129 *LBL F
130 RCL 01
131 "Fa=?"
132 PROMPT
133 STO 01
134 RCL 10
135 "Fe=?"
136 PROMPT
137 STO 10
138 GTO 98
139 *LBL 86
140 SIZE?
141 RCL 07
142 E
143 +
144 X>Y?
145 PSIZE
```

```
146  FC? 55
147  GTO 00
148  ADV
149  "FROM-TO
     VALUE"
150  >"   UNIT"
151  PRA
152  *LBL 00
153  FS? 02
154  GTO 00
155  *LBL 95
156  "FROM=?"
157  PROMPT
158  X#0?
159  GTO 01
160  "NODE#0"
161  AVIEW
162  PSE
163  GTO 95
164  *LBL 00
165  0
166  *LBL 01
167  FS? 55
168  ACX
169  E1
170  *
171  STO IND 07
172  XEQ IND 05
173  ASTO 09
174  ST+ IND 07
175  "TO=?"
176  PROMPT
177  CLA
178  ARCL X
179  >"  "
180  FS? 55
181  ACA
182  ST+ IND 07
183  FC? 02
184  GTO 00
185  E1
186  ST* IND 07
187  "CNTL. V=?"
188  PROMPT
189  ST+ IND 07
190  STO 08
191  *LBL 00
192  ENG 3
193  "VALUE=?"
194  PROMPT
195  CLA
196  ARCL 05
197  FS? 02
198  >">="
199  ARCL X
200  FS?C 03
201  1/X
202  FS? 02
203  CHS
204  STO IND 06
205  FIX 0
206  FS?C 02
207  GTO 00
208  >" "
209  ARCL 09
210  FC? 55
211  PROMPT
212  FC? 55
213  GTO 01
214  ACA
215  PRBUF
216  GTO 01
217  *LBL 00
218  FC? 55
219  GTO 01
220  PRBUF
221  3
222  SKPCHR
223  >"*V"
224  ARCL 08
225  ACA
226  PRBUF
227  *LBL 01
228  "COMP. #.="
229  ARCL 04
230  PROMPT
231  FS? 55
232  ACA
233  ADV
234  FS?C 00
235  GTO 00
236  2
237  ST+ 06
238  ST+ 07
239  SF 04
240  E
241  ST+ 04
242  GTO 87
243  *LBL 00
244  RCL 00
245  STO 04
246  RCL 03
247  X^2
248  RCL 04
249  GTO 10
250  *LBL 98
251  CF 22
252  ENG 3
253  "F="
254  ARCL 01
255  >" HZ"
256  AVIEW
257  PSE
258  E
259  FS?C 04
260  ST- 04
261  RCL 03
262  X^2
263  ST+ X
264  21.021
265  +
266  STO 00
267  CLX
268  *LBL 90
269  STO IND 00
270  DSE 00
271  GTO 90
272  22.021
273  STO 00
274  RCL 03
275  X^2
276  ST+ X
277  ST+ 00
278  RCL 04
279  ST+ X
280  +
281  E3
282  /
283  ST+ 00
284  *LBL 91
285  RCL IND 00
286  STO 09
287  ISG 00
288  0
289  RCL IND 00
290  INT
291  E1
292  /
293  STO 06
294  INT
295  STO 05
296  STO 07
297  ST- 06
```

```
298 E1
299 ST* 06
300 RCL IND 00
301 FRC
302 X=0?
303 GTO 00
304 RCL 05
305 XEQ 96
306 RCL 06
307 X=0?
308 GTO 01
309 STO 07
310 XEQ 96
311 -1
312 ST* 09
313 RCL 06
314 STO 07
315 RCL 05
316 XEQ 96
317 *LBL 00
318 RCL 05
319 STO 07
320 RCL 06
321 XEQ 96
322 *LBL 01
323 ISG 00
324 GTO 91
325 2
326 RCL 03
327 X<=Y?
328 GTO 00
329 E
330 -
331 E-3
332 +
333 STO 00
334 *LBL 94
335 XEQ 99
336 DSE 00
337 GTO 94
338 *LBL 00
339 E
340 0
341 XEQ 93
342 CHS
343 X<>Y
344 CHS
345 X<>Y
346 E
347 E
348 XEQ 93
349 XROM "/C"
350 R-P
351 LOG
352 20
353 *
354 ADV
355 AVIEW
356 FIX 3
357 "IGI="
358 ARCL X
359 >" dB"
360 PROMPT
361 "aG="
362 ARCL Y
363 FS? 43
364 >" RAD"
365 FC? 43
366 "` DEG"
367 PROMPT
368 RCL 02
369 FS? 01
370 ST+ 01
371 FC? 01
372 ST* 01
373 RCL 10
374 RCL 01
375 X<=Y?
376 GTO 98
377 ADV
378 "READY"
379 AVIEW
380 STOP
381 GTO F
382 *LBL 99
383 RCL 00
384 INT
385 E
386 -
387 E3
388 /
389 STO 05
390 *LBL 88
391 RCL 00
392 INT
393 E
394 -
395 E3
396 /
397 STO 06
398 *LBL 89
399 RCL 05
400 RCL 00
401 XEQ 93
402 RCL 00
403 RCL 06
404 XEQ 93
405 XROM "*C"
406 RCL 00
407 ENTER^
408 XEQ 93
409 XROM "/C"
410 RCL 05
411 RCL 06
412 XEQ 93
413 RDN
414 RDN
415 XROM "-C"
416 STO IND 07
417 E
418 ST+ 07
419 RDN
420 RDN
421 STO IND 07
422 ISG 06
423 GTO 89
424 ISG 05
425 GTO 88
426 RTN
427 *LBL 93
428 STO 07
429 RDN
430 RCL 03
431 *
432 RCL 07
433 +
434 ST+ X
435 22
436 +
437 STO 07
438 E
439 +
440 RDN
441 RCL IND T
442 RCL IND 07
443 RTN
444 *LBL "<VCS"
445 FC? 55
446 GTO 00
447 RCL 19
448 ACSPEC
449 RCL 20
450 ACSPEC
```

```
451 RCL 21
452 ACSPEC
453 0
454 RTN
455 *LBL "R="
456 "OHM"
457 FC? 55
458 GTO 00
459 RCL 16
460 ACSPEC
461 RCL 17
462 ACSPEC
463 RCL 18
464 ACSPEC
465 0
466 17
467 BLDSPEC
468 CLA
469 ARCL X
470 470*LBL 00
471 .1
472 RTN
473 *LBL "C="
474 FC? 55
475 GTO 00
476 RCL 14
477 ACSPEC
478 RCL 15
479 ACSPEC
480 RCL 14
481 ACSPEC
482 482*LBL 00
483 "F"
484 .2
485 RTN
486 *LBL "L="
487 FC? 55
488 GTO 00
489 RCL 11
490 ACSPEC
491 RCL 12
492 ACSPEC
493 RCL 13
494 ACSPEC
495 *LBL 00
496 "H"
497 .3
498 RTN
499 *LBL "/C"
500 R-P
501 1/X
502 RDN
503 CHS
504 GTO 00
505 *LBL "*C"
506 R-P
507 RDN
508 *LBL 00
509 RDN
510 R-P
511 ST* Z
512 RDN
513 ST+ Z
514 RDN
515 P-R
516 RTN
517 *LBL "-C"
518 ST- Z
519 RDN
520 ST- Z
521 RDN
522 RTN
523 *LBL 96
524 E
525 -
526 RCL 07
527 E
528 -
529 RCL 03
530 *
531 +
532 ST+ X
533 22
534 +
535 STO 08
536 RCL 09
537 .1
538 RCL IND 00
539 FRC
540 X<=Y?
541 GTO 00
542 E
543 ST+ 08
544 RCL 09
545 RCL 01
546 PI
547 *
548 *
549 ST+ X
550 .2
551 RCL IND 00
552 FRC
553 X=Y?
554 GTO 00
555 1/X
556 CHS
557 *LBL 00
558 RCL Z
559 ST+ IND 08
560 END
```

Network Analysis Algorithms for Programmable Calculators.

Dieter Lange, Vieweg book , ISBN: 3-528-04198-6

```
1    *LBL "INET"
2    SF 08
3    GTO 00
4    *LBL "UNET"
5    CF 08
6    *LBL 00
7    SF 27
8    CF 29
9    "N=?"
10   PROMPT
11   ENTER^
12   X^2
13   LASTX
14   3
15   *
16   +
17   55
18   +
19   SIZE?
20   X<>Y
21   X>Y?
22   PSIZE
23   RDN
24   RDN
25   STO 12
26   ΣREG 14
27   *LBL "LIST"
28   *LBL e
29   SF 00
30   XROM "*LST"
31   24.9
32   STO 20
33   " R C  L X BR"
34   SF 21
35   AVIEW
36   *LBL 23
37   ISG 20
38   RCL IND 20
39   INT
40   X=0?
41   GTO 27
42   ISG 20
43   RCL IND 20
44   *LBL 26
45   CF 07
46   CLA
47   ARCL IND Y
48   >"="
49   ENG 2
50   ARCL X
51   AVIEW
52   GTO 23
53   *LBL E
54   E1
55   /
56   ISG 20
57   STO IND 20
58   *LBL 27
59   SF 07
60   RCL IND 20
61   X=0?
62   GTO 14
63   "BRANCH "
64   E1
65   *
66   FIX 1
67   ARCL X
68   AVIEW
69   GTO 23
70   *LBL 14
71   RCL 20
72   INT
73   RCL 12
74   ST+ X
75   5
76   +
77   STO 24
78   -
79   2.9
80   +
81   STO 23
82   "READY"
83   PROMPT
84   GTO 14
85   *LBL A
86   E
87   GTO 13
88   *LBL a
89   2
90   GTO 13
91   *LBL B
92   3
93   GTO 13
94   *LBL b
95   4
96   GTO 13
97   *LBL C
98   5
99   GTO 13
100  *LBL c
101  6
102  GTO 13
103  *LBL D
104  7
105  GTO 13
106  *LBL d
107  8
108  GTO 13
109  *LBL "U0"
110  9
111  GTO 13
112  *LBL "I0"
113  E1
114  GTO 13
115  *LBL "a0"
116  11
117  *LBL 13
118  FS? 07
119  CHS
120  ISG 20
121  STO IND 20
122  X<>Y
123  ISG 20
124  STO IND 20
125  GTO 26
126  *LBL H
127  3
128  FS? 07
129  ST- 20
130  FS? 07
131  GTO 23
132  4
133  RCL T
134  X<0?
```

```
135 RDN
136 X<>Y
137 ST- 20
138 GTO 23
139 *LBL F
140 ST+ X
141 PI
142 *
143 *LBL G
144 "W="
145 ENG 2
146 ARCL X
147 STO 13
148 PROMPT
149 *LBL I
150 XROM "*BD"
151 *LBL "BN"
152 FS? 04
153 10^X
154 STO 13
155 SF 06
156 GTO 32
157 *LBL J
158 CF 06
159 159*LBL 32
160 RCL 12
161 XEQ 47
162 INT
163 3
164 +
165 E3
166 /
167 E
168 XEQ 47
169 INT
170 +
171 0
172 *LBL 30
173 STO IND Y
174 ISG Y
175 GTO 30
176 SF 03
177 CF 14
178 24.9
179 STO 20
180 *LBL 31
181 ISG 20
182 RCL IND 20
183 ISG 20
184 RCL IND 20
185 XEQ IND Y
186 FC? 14
187 GTO 31
188 GTO 60
189 *LBL 00
190 FC?C 03
191 XEQ 40
192 CF 15
193 FC? 08
194 SF 15
195 E
196 ST- 20
197 RCL IND 20
198 X=0?
199 SF 14
200 CLs
201 X<0?
202 SF 15
203 ABS
204 E1
205 *
206 RCL X
207 INT
208 STO 21
209 -
210 E1
211 *
212 CF 09
213 X=0?
214 SF 09
215 STO 22
216 CF 10
217 SF 02
218 RTN
219 *LBL 01
220 XEQ 71
221 ST+ 18
222 RTN
223 *LBL 02
224 XEQ 70
225 1/X
226 ST+ 18
227 RTN
228 *LBL 03
229 XEQ 71
230 GTO 90
231 *LBL 04
232 XEQ 70
233 RCL 13
234 *
235 ST+ 19
236 RTN
237 *LBL 05
238 XEQ 71
239 RCL 13
240 *
241 ST+ 19
242 RTN
243 *LBL 06
244 XEQ 70
245 *LBL 90
246 RCL 13
247 *
248 X=0?
249 E-30
250 1/X
251 ST- 19
252 RTN
253 *LBL 07
254 XEQ 71
255 ST+ 19
256 RTN
257 *LBL 08
258 XEQ 70
259 1/X
260 ST- 19
261 RTN
262 *LBL 09
263 SF 10
264 XEQ 71
265 ST+ 14
266 RTN
267 *LBL 10
268 SF 10
269 XEQ 70
270 ST+ 14
271 RTN
272 *LBL 11
273 RCL Z
274 ST- 14
275 P-R
276 ST+ 14
277 RDN
278 ST+ 15
279 RTN
280 *LBL 70
281 FS?C 01
282 GTO 72
283 CF 02
284 RTN
285 *LBL 71
```

```
FS? 01
CF 02
FS? 01
RTN
SF 01
*LBL 72
FS?C 02
RTN
STO 16
RCL 19
X^2
RCL 18
X^2
+
X=0?
GTO 73
ST/ 18
CHS
ST/ 19
GTO 74
*LBL 73
RDN
E30
STO 18
*LBL 74
RCL 15
RCL 14
RCL 19
RCL 18
XROM "C*"
STO 14
RDN
STO 15
RCL 16
RTN
*LBL 40
FS? 08
XEQ 71
FC? 08
XEQ 70
RCL 21
XEQ 47
RCL 19
ST+ IND Y
ISG Y
RCL 18
ST+ IND Z
FS? 09
GTO 43
RCL 22
RCL 21
X>Y?
X<>Y
XEQ 46
+
RCL 19
FS? 15
ST- IND Y
FC? 15
ST+ IND Y
ISG Y
RCL 18
FS? 15
ST- IND Z
FC? 15
ST+ IND Z
RCL 22
XEQ 47
RCL 19
ST+ IND Y
ISG Y
RCL 18
ST+ IND Z
*LBL 43
FC? 10
RTN
RCL 12
E
+
RCL 21
XEQ 46
+
RCL 15
ST+ IND Y
ISG Y
RCL 14
ST+ IND Z
FS? 09
RTN
RCL 12
E
+
RCL 22
XEQ 46
+
RCL 15
FS? 15
CHS
ST+ IND Y
ISG Y
RCL 14
FS? 15
CHS
ST+ IND Z
RTN
*LBL 46
STO Z
-
ST+ X
X<>Y
*LBL 47
RCL 24
RCL Y
-
*
RCL 23
+
RTN
*LBL 60
CF 00
RCL 12
E
-
E3
/
STO 20
ISG 20
GTO 61
SF 00
*LBL 61
RCL 20
INT
RCL X
XEQ 47
INT
STO 14
CLX
E
+
XEQ 47
STO 17
INT
3
-
E3
/
ST+ 14
RCL 14
STO 15
*LBL 62
ISG 15
ISG 15
GTO 63
```

```
439  FS? 00
440  GTO 63
441  ISG 20
442  GTO 61
443  SF 00
444  GTO 61
445  *LBL 63
446  RCL 15
447  .002
448  +
449  STO 16
450  RCL IND 14
451  ISG 14
452  RCL IND 14
453  STO Z
454  X^2
455  X<>Y
456  STO T
457  X^2
458  +
459  ST/ Y
460  CHS
461  ST/ Z
462  RDN
463  RCL IND 16
464  ISG 16
465  RCL IND 16
466  XROM "C*"
467  FS? 00
468  GTO 67
469  STO 18
470  X<>Y
471  STO 19
472  E
473  ST- 14
474  ST- 16
475  *LBL 66
476  RCL IND 16
477  ISG 16
478  RCL IND 16
479  RCL 19
480  RCL 18
481  XROM "C*"
482  X<>Y
483  ST- IND 17
484  ISG 17
485  X<>Y
486  ST- IND 17
487  ISG 17
488  ISG 16
489  GTO 66
490  GTO 62
491  *LBL 67
492  R-P
493  FS? 06
494  GTO 68
495  FS? 08
496  "I="
497  FC? 08
498  "U="
499  ENG 2
500  ARCL X
501  SF 21
502  AVIEW
503  "PHI="
504  ARCL Y
505  PROMPT
506  "F=?"
507  PROMPT
508  GTO F
509  *LBL 68
510  FS? 05
511  X<>Y
512  FS? 05
513  RTN
514  FC? 04
515  RTN
516  LOG
517  20
518  *
519  END
```

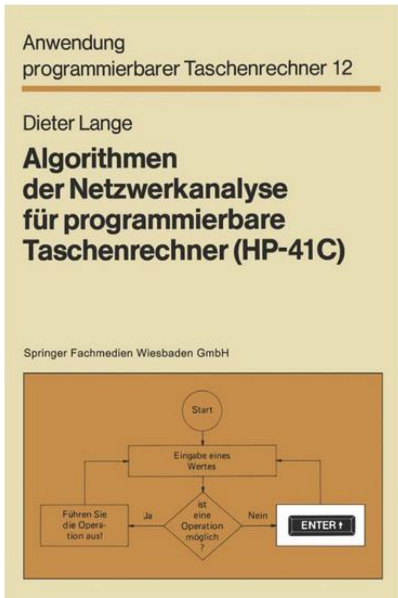
```
1   *LBL "RED"
2   DEG
3   CLRG
4   SF 27
5   CF 29
6   *LBL e
7   CF 00
8   XROM "*LST"
9   25.9
10  STO 25
11  "INPUT:"
12  SF 21
13  AVIEW
14  *LBL 70
15  ISG 25
16  13
17  RCL IND 25
18  X=0?
19  STOP
20  ABS
21  X>Y?
22  GTO 71
23  ISG 25
24  RCL IND 25
25  X<>Y
26  *LBL 71
27  SF 08
28  CLA
29  ARCL IND X
30  14
31  X<=Y?
32  GTO 72
33  CF 08
34  ENG 2
35  "`="
36  ARCL Z
37  *LBL 72
38  AVIEW
39  GTO 70
40  *LBL A
41  E
42  GTO 13
43  *LBL a
44  2
45  GTO 13
46  *LBL B
47  3
48  GTO 13
49  *LBL b
50  4
51  GTO 13
52  *LBL C
53  5
54  GTO 13
55  *LBL c
56  6
57  GTO 13
58  *LBL D
59  7
```

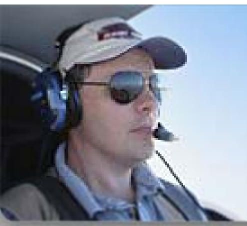
```
GTO 13
*LBL d
8
GTO 13
*LBL "U1"
9
GTO 13
*LBL "I1"
E1
GTO 13
*LBL "<)1"
11
GTO 13
*LBL "P"
17
GTO 13
*LBL "S"
19
GTO 13
*LBL "ST"
18
GTO 13
*LBL "RC"
16
GTO 13
*LBL "U^"
20
GTO 13
*LBL "I^"
21
GTO 13
*LBL "Z^"
22
GTO 13
*LBL E
24
96*LBL 13
FS? 08
CHS
ISG 25
STO IND 25
ABS
13
X<>Y
X>Y?
GTO 71
RCL Z
ISG 25
STO IND 25
X<>Y
GTO 71
*LBL H
FS? 08
GTO 75
E
ST- 25
*LBL 75
2
ENTER^
3
RCL IND 25
X<0?
RDN
X<>Y
ST- 25
GTO 70
*LBL I
XROM "*BD"
*LBL "BR"
FS? 04
10^X
STO 12
SF 06
*LBL 59
XEQ 65
25.9
STO 25
CF 07
*LBL 60
FS? 07
RTN
13
ISG 25
RCL IND 25
ABS
X>Y?
GTO 61
ISG 25
RCL IND 25
X<>Y
*LBL 61
X<>Y
XEQ IND Y
GTO 60
*LBL 01
XEQ 51
ST+ 15
RTN
*LBL 02
XEQ 50
1/X
ST+ 15
RTN
*LBL 03
XEQ 51
GTO 00
*LBL 06
XEQ 50
*LBL 00
RCL 12
*
X=0?
E-30
1/X
ST- 16
RTN
*LBL 04
XEQ 50
GTO 00
*LBL 05
XEQ 51
*LBL 00
RCL 12
*
ST+ 16
RTN
*LBL 07
XEQ 51
ST+ 16
RTN
*LBL 08
XEQ 50
1/X
ST- 16
RTN
*LBL 09
XEQ 51
ST+ 13
RTN
*LBL 10
XEQ 50
ST+ 13
RTN
*LBL 11
RCL T
ST- 13
P-R
ST+ 13
RDN
```

```
209 ST+ 14
210 RTN
211 *LBL 24
212 XEQ 62
213 *LBL 65
214 SF 02
215 0
216 STO 13
217 STO 14
218 STO 15
219 STO 16
220 RTN
221 *LBL 17
222 XEQ 50
223 *LBL 30
224 RCL 19
225 ST+ 15
226 RCL 20
227 ST+ 16
228 RCL 17
229 ST+ 13
230 RCL 18
231 ST+ 14
232 RTN
233 *LBL 19
234 XEQ 51
235 XEQ 63
236 CF 01
237 XEQ 51
238 GTO 30
239 *LBL 16
240 XEQ 62
241 CF 01
242 RCL 21
243 STO 13
244 RCL 22
245 STO 14
246 RCL 23
247 STO 15
248 RCL 24
249 STO 16
250 RTN
251 *LBL 18
252 XEQ 50
253 RCL 13
254 STO 21
255 RCL 14
256 STO 22
257 RCL 15
258 STO 23
259 RCL 16
260 STO 24
261 RTN
262 *LBL 40
263 R-P
264 FS? 06
265 GTO 41
266 SF 21
267 ARCL X
268 AVIEW
269 "PHI="
270 ARCL Y
271 PROMPT
272 "W=?"
273 PROMPT
274 GTO G
275 *LBL 41
276 SF 07
277 FS? 05
278 X<>Y
279 FS? 05
280 RTN
281 FC? 04
282 RTN
283 LOG
284 20
285 *
286 RTN
287 287*LBL 20
288 XEQ 51
289 "U="
290 GTO 00
291 *LBL 21
292 XEQ 50
293 "I="
294 *LBL 00
295 RCL 14
296 RCL 13
297 GTO 40
298 *LBL 22
299 XEQ 51
300 RCL 16
301 RCL 15
302 "Z="
303 GTO 40
304 *LBL F
305 ST+ X
306 PI
307 *
308 *LBL G
309 STO 12
310 "W="
311 ENG 2
312 ARCL X
313 AVIEW
314 *LBL J
315 CF 06
316 GTO 59
317 *LBL 50
318 FS?C 01
319 GTO 52
320 CF 02
321 RTN
322 *LBL 51
323 FS? 01
324 CF 02
325 FS? 01
326 RTN
327 SF 01
328 *LBL 52
329 FS?C 02
330 RTN
331 RCL 16
332 X^2
333 RCL 15
334 X^2
335 +
336 X=0?
337 GTO 53
338 ST/ 15
339 CHS
340 ST/ 16
341 GTO 54
342 *LBL 53
343 RDN
344 E30
345 STO 15
346 *LBL 54
347 RDN
348 RCL 14
349 RCL 15
350 *
351 RCL 13
352 RCL 16
353 *
354 +
355 RCL 13
356 RCL 15
357 *
358 STO 13
359 RDN
360 RCL 14
361 RCL 16
```

```
362  *
363  ST- 13
364  RDN
365  STO 14
366  RDN
367  RTN
368  *LBL 62
369  XEQ 50
370  *LBL 63
371  RCL 13
372  X<> 17
373  STO 13
374  RCL 14
375  X<> 18
376  STO 14
377  RCL 15
378  X<> 19
379  STO 15
380  RCL 16
381  X<> 20
382  STO 16
383  END
```

```
1   LBL "C*"
2   STO L
3   X<> Z
4   ST* L
5   RDN
6   ST* T
7   X<> L
8   X<> Z
9   ST* L
10  ST* Y
11  X<> L
12  ST- Z
13  RDN
14  ST+ Z
15  RDN
16  RTN
17  *LBL "*BD"
18  DEG
19  CF 04
20  "BODE? Y/N"
21  AVIEW
22  Y/N?
23  SF 04
24  "PHI=?"
25  PROMPT
26  SF 05
27  X=0?
28  CF 05
29  0
30  STO 03
31  XROM "PRPLOT"
32  STOP
33  RTN
34  *LBL "*LST"
35  "RS"
36  ASTO 01
37  "RP"
38  ASTO 02
39  "CS"
40  ASTO 03
41  "CP"
42  ASTO 04
43  "LS"
44  ASTO 05
45  "LP"
46  ASTO 06
47  "XS"
48  ASTO 07
49  "XP"
50  ASTO 08
51  "U"
52  ASTO 09
53  "I"
54  ASTO 10
55  "PHI"
56  ASTO 11
57  FS? 00
58  RTN
59  "ENTER"
60  ASTO 24
61  "PARLEL"
62  ASTO 17
63  "SERIAL"
64  ASTO 19
65  "RCL"
66  ASTO 16
67  "STO"
68  ASTO 18
69  "OUT U"
70  ASTO 20
71  "OUT I"
72  ASTO 21
73  "OUT Z"
74  ASTO 22
75  END
```





Op-Amp Gain and Offset Design with the HP-41C Programmable Calculator

May 26, 2009

This is a program I originally wrote for the HP-67 calculator, and then ported to the HP-41C series. This program is for designing offset-and-gain stages using a single operational amplifier. Such stages are often necessary to convert an input signal covering one range of voltages (e.g., 0.1V to 0.2V from a sensor) to an output signal covering a different range (e.g., 1.0 to 4.0V into an A/D converter).

Mathematically, such a stage performs a linear transformation on the input voltage,

$$V_{OUT} = m\ V_{IN} + b$$

where *m* is the *slope* and *b* is the *intercept* or *offset*.

Given an input voltage range, V_{IL} to V_{IH}, and an output voltage range, V_{OL} to V_{OH}, the slope and offset are given by:

$$m = \frac{V_{OH} - V_{OL}}{V_{IH} - V_{IL}}$$

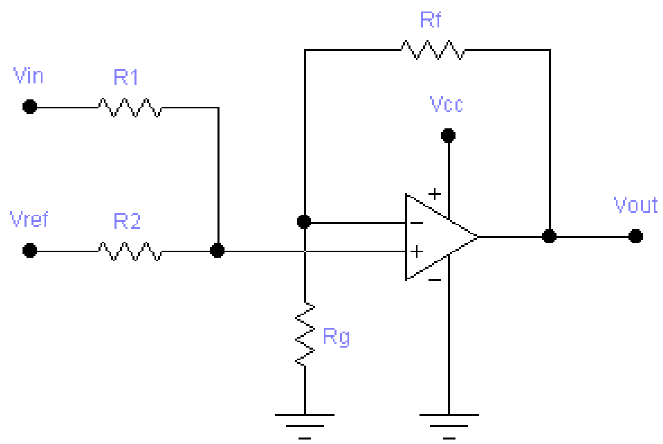
$$b = V_{OH} - m \cdot V_{IH}$$

There are four main cases to consider, with different circuits for each, that are addressed by this program:

1. Positive slope and offset ($m > 0$ and $b > 0$)
2. Positive slope and negative offset ($m > 0$ and $b < 0$)
3. Negative slope and positive offset ($m < 0$ and $b > 0$)
4. Negative slope and offset ($m < 0$ and $b < 0$)

Positive Slope and Offset

A positive slope and offset stage is implemented by the following circuit:



Positive gain, positive offset amplifier

The designer must select values for R_1 and R_F, and then appropriate values for R_2 and R_G can be calculated using the following formulae:

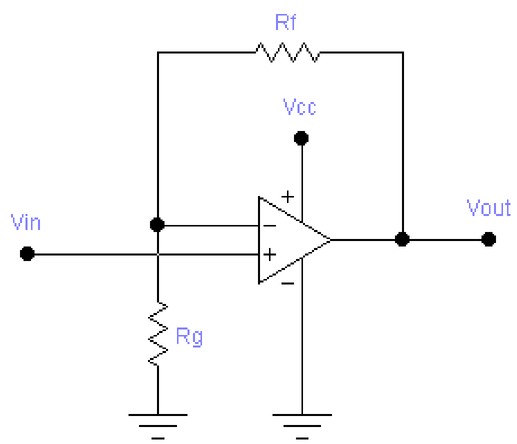
$$R_2 = \frac{m \cdot V_{REF} \cdot R_1}{b}$$

$$R_G = \frac{V_{REF} \cdot R_F}{(m-1) \cdot V_{REF} + b}$$

After determining theoretically ideal values for R_2 and R_G, real-world values can be chosen and the following formula applied to V_{IL} and V_{IH} to see the resulting values of V_{OL} and V_{OH} respectively:

$$V_{OUT} = \frac{(V_{IN} \cdot R_2 + V_{REF} \cdot R_1) \cdot \left(1 + \frac{R_F}{R_G}\right)}{R_1 + R_2}$$

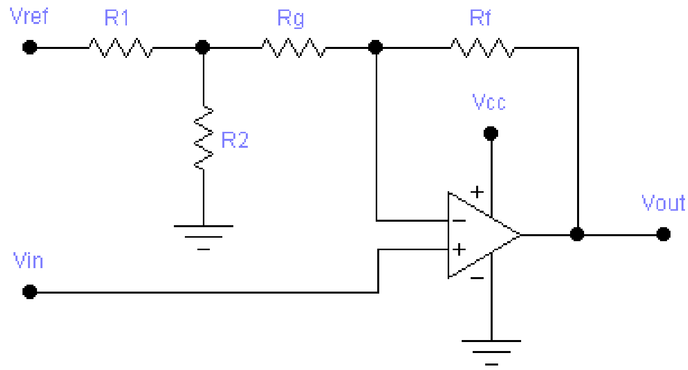
This case is also used to handle the special case where $b = 0$ (postive gain with no offset). In such a case, the formula for R_2 would result in dividing by zero, which means R_2 is infinite. In other words, R_2 and V_{REF} are not needed. The value of R_1 won't matter then either, and can be replaced by a direct connection. The circuit reduces to:



Positive gain, zero offset amplifier

Positive Slope and Negative Offset

The following circuit implements a positive slope and negative offset stage:



Positive gain, negative offset amplifier

After choosing values for R_1 and R_F, values for R_2 and R_G can be be calculated using these formulae:

$$R_2 = \frac{-b \cdot R_1}{(m-1) \cdot V_{REF} + b}$$

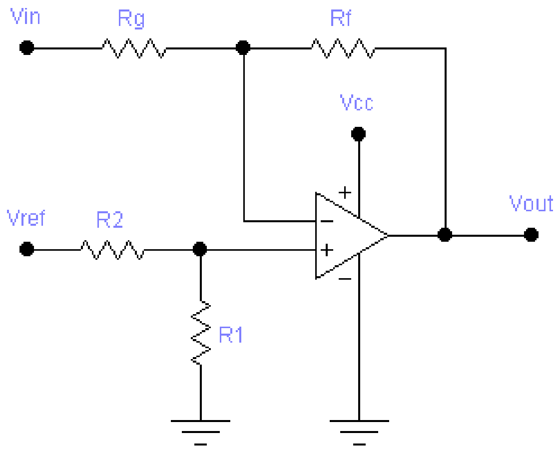
$$R_G = \frac{b \cdot R_1 + V_{REF} \cdot R_F}{(m-1) \cdot V_{REF}}$$

The following formula can then be used to determine the effect of real-world values of R_2 and R_G on the transformation:

$$V_{OUT} = \frac{\left(R_F + R_G + \frac{R_1 \cdot R_2}{R_1 + R_2}\right) \cdot V_{IN}}{R_G + \frac{R_1 \cdot R_2}{R_1 + R_2}} - \frac{V_{REF} \cdot R_2 \cdot R_F}{(R_1 + R_2) \cdot \left(R_G + \frac{R_1 \cdot R_2}{R_1 + R_2}\right)}$$

Negative Slope and Positive Offset

A negative slope and postive offset stage is implemented by the following circuit:



Negative gain, positive offset amplifier

Given values for R_1 and R_F, values for R_2 and R_G can be be calculated as follows:

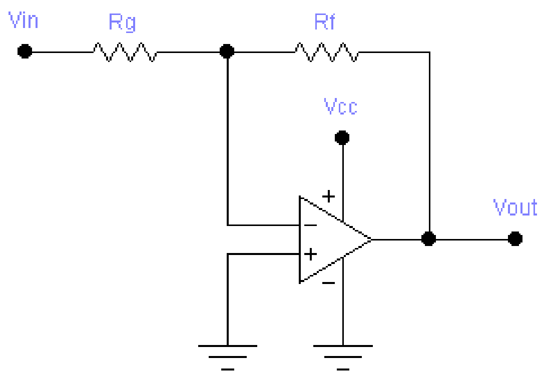
$$R_2 = \frac{\left((m-1)\cdot V_{REF} + b\right)\cdot R_1}{-b}$$

$$R_G = \frac{-R_F}{m}$$

The following formula can then be used to determine the effect of substituting real-world values of R_2 and R_G:

$$V_{OUT} = \frac{\left(R_G + R_F\right)\cdot V_{REF}\cdot R_1}{\left(R_1 + R_2\right)\cdot R_G} - \frac{V_{IN}\cdot R_F}{R_G}$$

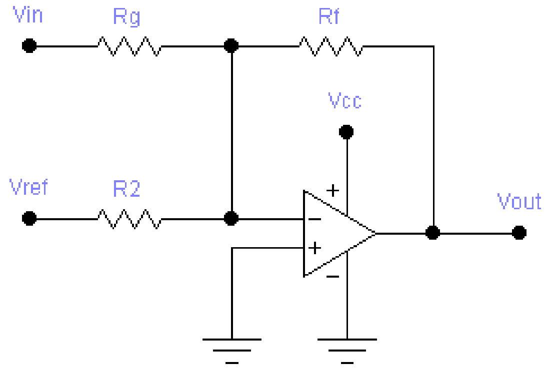
This case is also used to handle the special case where $b = 0$ (negative gain with no offset). As in case 1, the formula for R_2 would involve division by zero, so R_2 and V_{REF} are not needed. The non-inverting input of the op-amp can be connected directly to ground, giving the following circuit:



Negative gain, zero offset amplifier

Negative Slope and Offset

The following circuit implements a negative slope and offset stage:



Negative gain, zero offset amplifier

This circuit has no R_1, so it is only necessary to choose a value for R_F, after which R_2 and R_G are given by:

$$R_2 = \frac{-V_{REF} \cdot R_F}{b}$$

$$R_G = \frac{-R_F}{m}$$

The effect of using real-world values of R_2 and R_G can then be tested using this formula:

$$V_{OUT} = \frac{-V_{IN} \cdot R_F}{R_G} - \frac{V_{REF} \cdot R_F}{R_2}$$

Limitations

Theoretically, the formulae presented here work perfectly well for gains between -1 and 1 (i.e. $|m| < 1$). However, many real-world op-amps are unstable in such cases. Instead, it will usually be necessary to design for a higher gain ($|m| \geq 1$), with an attenuator on the input side. The design procedures for this are described on Texas Instruments' *Op Amp Gain and Offset Page*.

Using the Program

First type in the program and save it, or read it from a previously recorded magnetic card. The card should be labelled as follows:

OP-AMP GAIN AND OFFSET STAGE DESIGN

V_{REF}	V_{IL},V_{IH}	V_{OL},V_{OH}	
→m,b	→CASE	R_1,R_F→R_2,R_G	R_2,R_G→V_{OL},V_{OH}

Forward Solution: Finding R_2 and R_G

Consider the following example. A sensor has an output ranging from 0.5V to 0.7V, and we want to interface it to an A/D converter that is expecting an input between 1V and 4V. There is no reference voltage available other than the well regulated 5V supply voltage of the circuit. Use a 10kΩ resistor for R_1, and 100kΩ for R_F.

Follow these steps to solve the problem:

Description	Keystrokes	Display
Use engineering notation	ENG 2	0.00 00
Enter V_{REF}	5 a	5.00 00
Enter V_{IL} and V_{IH}	0.5 ENTER 0.7 b	500. -03
Enter V_{OL} and V_{OH}	1 ENTER 4 c	1.00 00
Compute slope m	A	15.0 00
Compute offset b	R/S	-6.50 00
Determine case number	B	2.00 00
Enter R_1 and R_F, compute R_2	10 EEx 3 ENTER 100 EEx 3 C	1.02 03
Compute R_G	R/S	6.21 03

Notes

It is not necessary to compute the slope and offset (by pressing `A`) before determining the case number. Likewise, it isn't necessary to determine the case number (by pressing `B`) before computing resistor values (although you'll want to know the case number in order to know which circuit to build). The program keeps track of which information is up to date, and will (re)compute anything that it needs that hasn't already been computed.

Reverse Solution: Finding the Effect of R_2 and R_G on V_{OL} and V_{OH}

The closest available 5% resistor values for R_2 and R_G are 1kΩ and 6.2kΩ respectively. What effect does using these have on the solution? Follow these steps to find out:

Description	Keystrokes	Display
Enter R_2 and R_G, compute V_{OL}	1 EEx 3 ENTER 6.2 EEx 3 D	1.13 00
Compute V_{OH}	R/S	4.15 00

This is within the A/D's input range at the lower bound, but outside the range at the upper bound. What happens if we use the next available value for R_G, 6.8kΩ, instead?

Description	Keystrokes	Display
Enter R_2 and R_G, compute V_{OL}	1 EEx 3 ENTER 6.8 EEx 3 D	1.09 00
Compute V_{OH}	R/S	3.88 00

This is almost centered within the desired output range, and covers 93% of it.

The only remaining concern is how component tolerances might affect the solution. This can be analyzed by trying different combinations of R_1, R_2, R_F, and R_G representing resistors that are maximally out of tolerance (±5%) in each direction.

For example, to test the case where R_1 and R_F are 5% low and R_2 and R_G are 5% high, follow these steps:

Description	Keystrokes	Display
Enter low R_1 and R_F, ignore computed R_2	0.95 EEx 3 ENTER 95 EEx 3 C	920. 00
Enter high R_2 and R_G, compute V_{OL}	1.05 EEx 3 ENTER 7.14 EEx 3 D	808. -03
Compute V_{OH}	R/S	3.48 00

The results show that in this case, the lower limit of the output is out of range, meaning either a redesign is necessary, or tighter tolerance components are needed.

Cases where b = 0

In cases where the offset, b, is zero, the program will instead use $b = 10^{-9}$ as the offset. This will avoid any division-by-zero errors. The program will then use case 1 (if $m > 0$) or case 3 (if $m < 0$) to compute the solution. In both cases, the computed value for R_2 will be very large, typically around 10^9 times the value specified for R_1. This indicates that R_2 and V_{REF} can be omitted, and that R_1 can be replaced by a direct connection.

Program Listing

Line	Instruction	Comments
01♦	LBL "GOFF"	
02♦	LBL a	*Enter available reference voltage*
03	STO 09	
04	RTN	
05♦	LBL b	*Enter V_{IL} and V_{IH}*
06	STO 06	*Store V_{IH}*
07	x↔y	
08	STO 05	*Store V_{IL}*
09	CF 00	*Must recompute m and b*
10	RTN	
11♦	LBL c	*Enter V_{OL} and V_{OH}*
12	STO 08	*Store V_{OH}*
13	x↔y	
14	STO 07	*Store V_{OL}*
15	CF 00	*Must recompute m and b*
16	RTN	

17♦	LBL A	*Compute transfer function slope (m) and intercept (b)*
18	FS? 00	*Are m and b already up to date?*
19	GTO 05	
20	RCL 08	*Compute and store* $m = (V_{OH} - V_{OL}) \div (V_{IH} - V_{IL})$
21	RCL 07	
22	−	
23	RCL 06	
24	RCL 05	
25	−	
26	÷	
27	STO 20	
28	RCL 05	*Compute and store* $b = V_{OH} - m\ V_{IL}$
29	×	
30	RCL 07	
31	−	
32	CHS	
33	x≠0?	
34	GTO 07	
35	1 E-9	*Make sure b is never exactly 0;*
36♦	LBL 07	
37	STO 21	
38	SF 00	*Stored m and b are now up to date*
39	CF 01	*Must recompute case number now*
40♦	LBL 05	*Display computed or already up-to-date m and b*
41	RCL 20	
42	RTN	*Return, leaving m on stack*
43	RCL 21	*Display b if user pressed R/S after return*
44	RTN	
45♦	LBL B	*Compute case number and store in I*
46	XEQ A	*Make sure m and b are up to date*
47	FS? 01	*Is case number already up to date*
48	GTO 08	
49	x<0?	
50	GTO 06	*Handle cases where* $m < 0$
51	1	
52	STO 25	
53	RCL 21	
54	x<0?	
55	ISZ 25	*Case 2: m positive and b negative*
56	GTO 08	*Case 1: m positive and b positive*
57♦	LBL 06	*Cases where m is negative*
58	3	
59	STO 25	
60	RCL 21	
61	x<0?	
62	ISZ 25	*Case 4: m negative and b negative*

63♦	LBL 08	*Case 3: m negative and b positive*
64	SF 01	*Case number is now up to date (or was already up to date)*
65	RCL 25	*Display case number and return*
66	RTN	
67♦	LBL C	*Compute solution using appropriate case*
68	STO 03	*Store* R_F
69	x↔y	
70	STO 01	*Store* R_1
71	XEQ B	*Get case number*
72	GTO IND 25	*Branch to appropriate case*
73♦	LBL 01	*Positive m and positive b*
74	RCL 01	*Compute and store* $R_2 = V_{REF}\ R_1\ m \div b$
75	RCL 09	
76	×	
77	RCL 20	
78	×	
79	RCL 21	
80	÷	
81	STO 02	
82	R/S	*Display* R_2*; 9.99e99 means "open circuit"*
83	RCL 09	*Compute and store* $R_G = V_{REF}\ R_F \div (V_{REF}\ (m - 1) + b)$
84	RCL 03	
85	×	
86	XEQ 09	*Compute* $V_{REF}\ (m - 1) + b$
87	GTO 05	*Divide, store* R_G*, and return, leaving* R_G *on stack*
88♦	LBL 02	*Positive m and negative b*
89	RCL 01	*Compute and store* $R_2 = -R_1\ b \div (V_{REF}\ (m - 1) + b)$
90	RCL 21	
91	CHS	
92	×	
93	XEQ 09	*Compute* $V_{REF}\ (m - 1) + b$*, leaving* $V_{REF}\ (m - 1)$ *in register 0*
94	÷	
95	STO 02	
96	R/S	*Display* R_2
97	RCL 01	*Compute and store* $R_G = (R_1\ b + V_{REF}\ R_F) \div (V_{REF}\ (m - 1))$
98	RCL 21	
99	×	
100	RCL 09	
101	RCL 03	
102	×	
103	+	
104	RCL 00	*Subroutine 9 left* $V_{REF}\ (m - 1)$ *in register 0 for us*
105	GTO 05	*Divide, store* R_G*, and return, leaving* R_G *on stack*
106♦	LBL 03	*Negative m and positive b*
107	RCL 01	*Compute and store* $R_2 = R_1\ (V_{REF}\ (m - 1) + b) \div -b$
108	XEQ 09	*Compute* $V_{REF}\ (m - 1) + b$

109	×	
110	RCL 21	
111	CHS	
112	÷	
113	STO 02	
114	R/S	*Display R_2*
115	RCL 03	*Compute and store $R_G = -R_F \div m$*
116	CHS	
117	RCL 20	
118	GTO 05	*Divide, store R_G, and return, leaving R_G on stack*
119♦	LBL 04	*Negative m and negative b*
120	RCL 09	*Compute and store $R_2 = V_{REF}\, R_F \div -b$*
121	RCL 03	
122	×	
123	RCL 21	
124	CHS	
125	÷	
126	STO 02	
127	R/S	*Display R_2*
128	RCL 03	*Compute and store $R_G = R_F \div m$*
129	RCL 20	
130	CHS	
131♦	LBL 05	*Common code to compute $y \div x$ and store in R_G*
132	÷	
133	STO 04	
134	RTN	*Return, leaving R_G on stack*
135♦	LBL 09	*Compute $V_{REF}\,(m-1) + b$, leaving $V_{REF}\,(m-1)$ in register 0*
136	RCL 20	
137	1	
138	−	
139	RCL 09	
140	×	
141	STO 00	*Save $V_{REF}\,(m-1)$ in register 0 for later*
142	RCL 21	
143	+	
144	RTN	
145♦	LBL D	*Compute V_{OL},V_{OH} using supplied R_2,R_G*
146	STO 04	*Store R_G*
147	x↔y	
148	STO 02	*Store R_2*
149	XEQ B	*Get case number*
150	RCL 05	
151	XEQ IND 25	*Compute V_{OL} from V_{IL}*
152	R/S	*Display computed V_{OL}*
153	RCL 06	
154	GTO IND 25	*Compute V_{OH} from V_{IH}*

155♦	LBL 01	*Compute V_{OUT} for positive m and positive b*
156	RCL 02	
157	×	
158♦	LBL 00	*Entry point to compute $(x + V_{REF} R_1) (1 + R_F \div R_G) \div (R_1 + R_2)$*
159	RCL 09	
160	RCL 01	
161	×	
162	+	
163	RCL 03	
164	RCL 04	
165	÷	
166	1	
167	+	
168	×	
169	RCL 01	
170	RCL 02	
171	+	
172	÷	
173	RTN	
174♦	LBL 02	*Compute V_{OUT} for positive m and negative b*
175	RCL 01	
176	1/x	
177	RCL 02	
178	1/x	
179	+	
180	1/x	
181	RCL 04	
182	+	
183	STO 00	*Store $R_G + R_1 R_2 \div (R_1 + R_2)$ for later*
184	RCL 03	
185	+	
186	×	
187	RCL 00	
188	÷	
189	RCL 09	
190	RCL 02	
191	×	
192	RCL 03	
193	×	
194	RCL 01	
195	RCL 02	
196	+	
197	RCL 00	
198	×	
199	÷	
200	−	

201	RTN	
202♦	LBL 03	*Compute V_{OUT} for negative m and positive b*
203	0	
204	XEQ 00	*Compute $(V_{REF} R_1) (1 + R_F \div R_G) \div (R_1 + R_2)$*
205	GTO 09	*Subtract $V_{IN} R_F \div R_G$*
206♦	LBL 04	*Compute V_{OUT} for negative m and negative b*
207	RCL 09	
208	RCL 03	
209	×	
210	RCL 02	
211	÷	
212	CHS	
213♦	LBL 09	*Entry point to compute $x - y R_F \div R_G$*
214	x↔y	
215	RCL 03	
216	×	
217	RCL 04	
218	÷	
219	–	
220	RTN	

Two interesting aspects of this program are its use of indirect addressing for branching to the forward and reverse solution subroutines for each of the four cases, and its use of repeated labels.

The forward solution for each of the four slope/offset cases is implemented by a sequence of instructions starting with the label corresponding to the case number (1 to 4). When the user presses `C`, the case number is computed if necessary, and a `GTO (i)` instruction then branches to the appropriate case.

Similarly, the reverse solution for each case is also labeled according to the case number. When the user presses `D`, V_{IL} is recalled to the stack, after which a `GSB (i)` instruction causes V_{OL} to be calculated. Then V_{IH} is recalled, and `GTO (i)` is used to calculate V_{OH} and then return.

So, there are two each of `LBL 1` through `LBL 4`. This works because the HP-41C/CV/CX (and other vintage HP calculators) search forwards from the current step for the matching label. Thus the `GTO (i)` in step 73 will branch to the appropriate forward solution, and the `GSB (i)` in step 152 and `GTO (i)` in step 155 will branch to the appropriate reverse solution.

This program also makes use of many small subroutines to compute sub-expressions common to multiple solutions. Since there were not enough labels for all the subroutines needed, the same labels were used more than once. Had this not been done, the same sequence of steps would have been repeated several times, and the program would not have fit into the calculator's 224 step memory.

Registers and Flags

Register	Use
00	*Temporary register*
01	R_1 *(Ohms)*
02	R_2 *(Ohms)*
03	R_F *(Ohms)*
04	R_G *(Ohms)*
05,06	V_{IL} *and* V_{IH}, *input voltage range*
07,08	V_{OL} *and* V_{OH}, *output voltage range*
09	V_{REF}
11	*m, slope of transfer function*
12	*b, intercept of transfer function*
10	*Case number (1,2,3,4)*

Flag	Meaning
00	*m and b are up to date*
01	*Case number is up to date*

Revision History

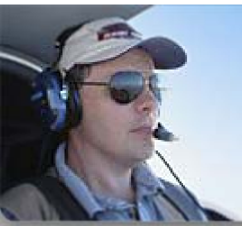
2009-May-26 — Initial release.

Like Be the first of your friends to like this.

Related Articles

If you've found this article useful, you may also be interested in:

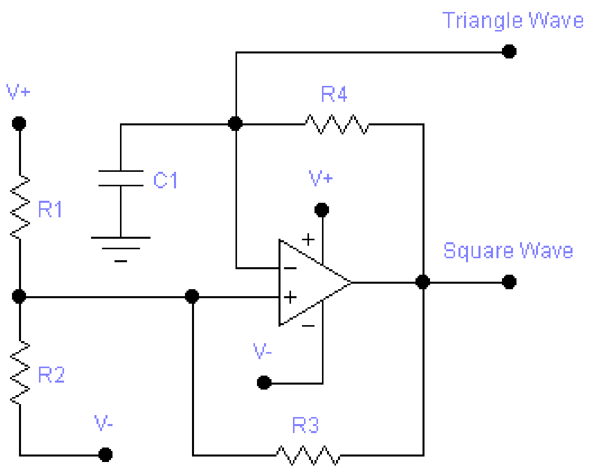
- *Op-Amp Oscillator Design with the HP-41C Programmable Calculator*
- *Low-Sensitivity Sallen-Key Filter Design with the HP-41C Programmable Calculator*
- *Low-Sensitivity Sallen-Key Filter Design with the HP-67 Programmable Calculator*
- *Op-Amp Gain and Offset Design with the HP-67 Programmable Calculator*
- *Op-Amp Oscillator Design with the HP-67 Programmable Calculator*
- *Resistor Network Solver for the HP-67 Programmable Calculator*
- *A Matrix Multi-Tool for the HP 35s Programmable Calculator*
- *Curve Fitting for the HP 35s Programmable Calculator*



Op-Amp Oscillator Design with the HP-41C Programmable Calculator

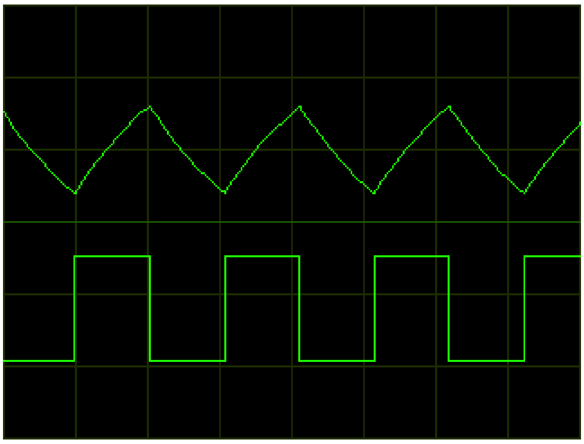
May 25, 2009

I originally wrote this program for the HP-67 calculator, and then ported it to the HP-41C series. This program selects component values for an op-amp based relaxation oscillator, given the desired frequency and output wave form peak voltages. It can also solve the inverse problem, finding the frequency and voltages resulting from given component values. The following is the schematic for such an oscillator:



R_1 and R_2 form a voltage divider, with an additional input from the op-amp output through R_3. When the op-amp output is at a high-level, the voltage at the non-inverting input of the op-amp is higher than when the op-amp output is at a low level. When the output is high, capacitor C_1 also charges through R_4 until the voltage across it (which is applied to the op-amp's inverting input) reaches the voltage at the non-inverting input. At that time, the op-amp output goes low, and the capacitor begins to discharge through R_4 until the voltage once again reaches the (now lower) voltage at the non-inverting input.

If one were to monitor the op-amp output, it would alternate between a high level (V_{OH}, generally close to positive supply voltage, V_{POS}) and a low level (V_{OL}, generally close to negative supply voltage, V_{NEG}). The duty cycle of this square wave depends on the relative time it takes to charge and discharge C_1 through R_4, which in turn depends on the low (V_{PL}) and high (V_{PH}) peak voltages that C_1 cycles between (which in turn depend on R_1, R_2, and R_3). Monitoring the voltage across C_1 shows a triangle wave.



With this program, you can select components for such an oscillator to achieve a desired frequency, and if it matters to your design, desired low and high triangle peaks (V_{PL} and V_{PH}). After the program computes the required component values, you can modify these values to match those actually available in the real world. The program will then compute what effect these changes have on the frequency and triangle peak voltages.

The following equations describe the operation of the oscillator:

$$\%DC = \frac{\ln\left(\frac{V_{OH} - V_{PL}}{V_{OH} - V_{PH}}\right)}{\ln\left(\frac{V_{OL} - V_{PH}}{V_{OL} - V_{PL}}\right) + \ln\left(\frac{V_{OH} - V_{PL}}{V_{OH} - V_{PH}}\right)}$$

$$R_2 = - \frac{R_1\left(\left(V_{OL} - V_{PL} - V_{OH} + V_{PH}\right) V_{NEG} + V_{OH} V_{PL} - V_{PH} V_{OL}\right)}{\left(V_{OL} - V_{PL} - V_{OH} + V_{PH}\right) V_{POS} + V_{OH} V_{PL} - V_{PH} V_{OL}}$$

$$R_3 = - \frac{R_1\left(\left(V_{OL} - V_{PL} - V_{OH} + V_{PH}\right) V_{NEG} + V_{OH} V_{PL} - V_{PH} V_{OL}\right)}{\left(V_{POS} - V_{NEG}\right)\left(V_{PL} - V_{PH}\right)}$$

$$R_4 = \frac{1}{C_1 f\left(\ln\left(\frac{V_{OL} - V_{PH}}{V_{OL} - V_{PL}}\right) + \ln\left(\frac{V_{OH} - V_{PL}}{V_{OH} - V_{PH}}\right)\right)}$$

You will first need to choose values for C_1 and R_1 arbitrarily, since for any desired frequency and given C_1 and R_1, it will be possible to find (possibly impractical) values for R_2, R_3, and R_4. A good choice for R_1 is generally somewhere around 10kΩ to 100kΩ. The choice of C_1 depends on the frequency, and a readily available value near (50/*f*) μF is usually suitable.

Using the Program

First type in the program and save it, or read it from a previously recorded magnetic card. The card should be labelled as follows:

OP-AMP OSCILLATOR DESIGN				
V_{NEG},V_{POS}	V_{OL},V_{OH}	C_1	R_1	V_{PL},V_{PH}
f	→%DC	→R_2→	→R_3→	→R_4→

Forward Solution: Finding R_2, R_3, and R_4

Consider the following example: It is desired to find values for R_2, R_3, and R_4 to produce an oscillator of about 2500Hz, with a triangle waveform that oscillates between 1.2V and 1.4V. The op-amp is to operate from a single 5V supply, and the op-amp's output is capable of a low of 0.3V and a high of 5V. Use a 0.022µF capacitor for C_1, and a 22kΩ resistor for R_1.

Follow these steps to solve the problem:

Description	Keystrokes	Display
Select engineering notation	ENG 2	0.00 00
Enter power supply voltages	0 ENTER 5 a	0.00 00
Enter low and high level output voltages	0.3 ENTER 5 b	300. -03
Enter C_1 (Farads)	0.022 EEx CHS 6 c	22.0 -09
Enter R_1 (Ohms)	22 EEx 3 d	22.0 03
Enter desired triangle lower and upper voltage peaks	1.2 ENTER 1.4 e	1.20 00
Enter desired frequency (Hz)	2500 A	2.50 03
Compute square wave duty cycle	B	788. -03
Compute value of R_2 (Ohms)	C	7.26 03
Compute value of R_3 (Ohms)	D	123. 03
Compute value of R_4 (Ohms)	E	71.4 03

Notes

Specifying V_{OL} and V_{OH} is optional. If this step is omitted, the program will assume the op-amp output can span the entire negative and positive supply voltage range.

If the triangle wave form peak voltages don't matter to your design (because you're only using the square wave output), you don't need to specify them. The program will assume peak voltages ranging from $V_{OL}+(V_{OH}-V_{OL})/3$ to $V_{OH}-(V_{OH}-V_{OL})/3$, which is the middle third of the op-amp output voltage range. This also happens to result in a 50% duty cycle.

To achive a low duty cycle square wave, choose V_{PL} and V_{PH} close to the bottom of the op-amp output range (V_{OL}). Likewise for a high duty cycle, choose V_{PL} and V_{PH} close to the top of the range (V_{OH}).

For the most stable oscillation frequency, choose V_{PL} and V_{PH} far away from the op-amp output voltage limits, V_{OL} and V_{OH} (to keep the triangle edges steep), and far away from each other (to keep any variations a small percentage of the overall voltage swing). Since these two goals are at odds with one another, a good compromise is to select V_{PL} and V_{PH} to span the middle of the V_{OL} to V_{OH} range (which is the default if V_{PL} and V_{PH} aren't specified).

Reverse Solution: Finding V_{PL}, V_{PH}, Frequency, and Duty Cycle

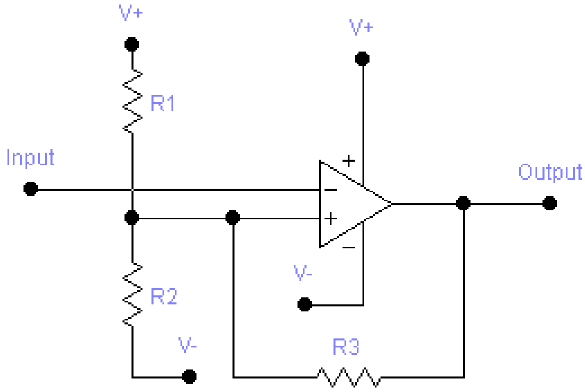
After finding the above ideal solution, we'll want to use real-world component values to build the physical circuit. The closest E-24 resistor values to those computed are: R_2 = 7.5kΩ, R_3 = 120kΩ, and R_4 = 68kΩ. What effect will using these values have on the frequency, V_{PL}, and V_{PH}?

These are the steps to find out:

Description	Keystrokes	Display
Enter new value for R_2	7.5 EEx 3 C	7.50 03
Enter new value for R_3	120 EEx 3 D	120. 03
Enter new value for R_4	68 EEx 3 E	68.0 03
Compute resulting value for V_{PL}	e	1.23 00
Compute resulting value for V_{PH}	R/S	1.44 00
Compute resulting frequency	A	2.57 03

Other Uses for this Program

The oscillator design facilitated by this program consists of two parts, a comparator with hysteresis, and a capacitor being charged and discharged by the comparator output through a resistor. The equations describing the comparator aspect of the circuit are not affected by those describing the behaviour of the resistor-capacitor network, so the program can be used to design such comparators for other applications.



Here is a brief example of using this program to design a comparator: Assume we want to design a comparator operating from a +/-12V supply, whose output goes low when the voltage exceeds +2V, and goes high when the voltage subsequently drops below -3V. Assume the op amp used has an output that can swing to within 0.7V of the voltage limits. Use a 10kΩ resistor for R_1. Follow these steps to solve the problem:

Description	Keystrokes	Display
Enter power supply voltages	12 CHS ENTER 12 ■ a	-12.0 00
Enter low and high level output voltages	11.3 CHS ENTER 11.3 ■ b	-11.3 00
Enter R_1	10 EEx 3 ■ d	10.0 03
Enter desired lower and upper switching points	3 CHS ENTER 2 ■ e	-3.00 00
Compute value of R_2	C	8.98 03
Compute value of R_3	D	16.7 03

Now select the closest real-world resistor values for R_2 and R_3 and determine how that affects the switching points:

Description	Keystrokes	Display
Enter new value for R_2	9.1 EEx 3 C	9.10 03
Enter new value for R_3	16 EEx 3 D	16.0 03
Compute resulting lower switching point	■ e	-3.03 00
Compute resulting upper switching point	R/S	2.16 00

Additional Real-World Considerations

The mathematical model used as the basis of this program assumes that V_{OL} and V_{OH} are constant, regardless of load. For sufficiently low current, this is close enough to true to be ignored. Thus it is important to use fairly high resistor values for R_3 and R_4 (10kΩ or bigger to be on the safe side). If the value of R_3 computed by the program is too low, start with a *higher* value for R_1. Similarly, if the value computed for R_4 is too low, use a *lower* value for C_1.

Some op-amps have an open-collector output. This means that when the output is low, it is pulled low through an output transistor, but when the output is high, it is simply floating. Thus, a pull-up resistor is needed to pull the output high. The chosen pull-up resistor must meet two requirements:

1. It must have a high-enough resistance that the output transistor can overcome the pull-up current when the output is low.

2. It must have a low-enough resistance that it is not so large a percentage of the resistance of R_3 or R_4 that it throws off the solution.

For the LM339 comparator that I often use in my designs, I've found that a 1kΩ resistor works well, together with R_3 and R_4 values about 100 times as much. As described above, use a higher value for R_1 to achieve a higher value for R_3, and use a lower value for C_1 to achieve a higher R_4.

Program Listing

Line	Instruction	Comments
01♦	LBL "OSC"	
02♦	LBL a	*Store V_{NEG} and V_{POS}*
03	STO 06	*V_{POS}*
04	x↔y	
05	STO 05	*V_{NEG}*
06	x↔y	*Fall through and initialize V_{OL} and V_{OH} to V_{NEG} and V_{POS}*
07♦	LBL b	*Store V_{OL} and V_{OH}*
08	CF 22	
09	STO 08	*V_{OH}*
10	x↔y	
11	STO 07	*V_{OL}*
12	–	*Initialize V_{PL} and V_{PH}*
13	3	
14	÷	*$(V_{OH}-V_{OL})/3$*
15	RCL 07	
16	x↔y	
17	+	
18	STO 11	*Set $V_{PL} = V_{OL} + (V_{OH}-V_{OL})/3$*
19	RCL 08	
20	LASTx	
21	–	
22	STO 12	*Set $V_{PH} = V_{OH} - (V_{OH}-V_{OL})/3$*
23	CF 01	
24	RCL 08	*Leave V_{OL} and V_{PH} on stack as feedback to user*
25	RCL 07	
26	RTN	

27♦	LBL c	*Store C_1*
28	CF 22	
29	STO 13	
30	RTN	
31♦	LBL d	*Store R_1*
32	CF 22	
33	STO 01	
34	RTN	
35♦	LBL e	*Store or compute (if necessary) V_{PL} and V_{PH}*
36	FS?C 22	*If data entered, store new V_{PL} and V_{PH}*
37	GTO 09	
38♦	LBL 01	*Otherwise, compute V_{PL} and V_{PH} if necessary*
39	FS? 01	*Need to compute V_{PL} and V_{PH}?*
40	GTO 08	
41	RCL 12	*Recall already-up-to-date V_{PL} and V_{PH}*
42	RCL 11	
43	RTN	
44	RCL 12	*If user presses R/S after seeing V_{PL}, display V_{PH}*
45	RTN	
46♦	LBL 08	*Recompute V_{PL} and V_{PH}*
47	RCL 01	
48	RCL 03	
49	×	
50	STO 14	
51	RCL 02	
52	RCL 03	
53	×	
54	STO 10	
55	+	
56	RCL 02	
57	RCL 01	
58	×	
59	STO 09	
60	+	
61	1/x	
62	STO 15	
63	RCL 05	
64	×	
65	RCL 14	
66	×	
67	RCL 15	
68	RCL 06	
69	×	
70	RCL 10	
71	×	
72	+	

73	STO 14	*Partial result common to V_{PL} and V_{PH}*
74	RCL 15	
75	RCL 09	
76	×	
77	STO 15	*End of computation common to V_{PL} and V_{PH}*
78	RCL 07	*Compute V_{PL}*
79	×	
80	+	*End of computation of V_{PL}*
81	RCL 15	*Compute V_{PH}*
82	RCL 08	
83	×	
84	RCL 14	
85	+	*End of computation of V_{PH}; V_{PL} is in Y-register*
86♦	LBL 09	*Store entered or computed V_{PH} and V_{PL}*
87	STO 12	*Store V_{PH}*
88	x↔y	
89	STO 11	*Store V_{PL}*
90	CF 01	*V_{PL} and V_{PH} are now up to date*
91	RTN	
92	RCL 12	*If user presses R/S after seeing V_{PL}, display V_{PH}*
93	RTN	
94♦	LBL B	*Compute duty cycle*
95	CF 22	
96	XEQ 07	*Get numerator and denominator (also used for computing R_4 or f)*
97	LASTx	*Numerator*
98	x↔y	
99	÷	
100	RTN	
101♦	LBL 07	*Compute denominator of duty cycle, leaving numerator in LSTx*
102	XEQ 01	*Recompute V_{PL} and V_{PH} if necessary; leaves V_{PH} in X, V_{PL} in Y*
103	RCL 08	*Compute first half of denominator*
104	−	
105	RCL 12	
106	RCL 08	
107	−	
108	÷	
109	LN	
110	RCL 12	*Compute second half of denominator (which is also the numerator)*
111	RCL 07	
112	−	
113	RCL 11	
114	RCL 07	
115	−	
116	÷	
117	LN	

118	+	*Combine two halves, leaving numerator in LSTx*
119	RTN	
120♦	LBL A	*Store or compute f*
121	FS?C 22	
122	GTO 00	
123	XEQ 07	*Get denominator (also used for computing R_4 and duty cycle)*
124	RCL 04	*Multiply by R_4 and C_1*
125	×	
126	RCL 13	
127	×	
128	1/x	
129♦	LBL 00	*Store entered or computed f*
130	STO 00	
131	RTN	
132♦	LBL C	*Store or compute R_2*
133	FS?C 22	
134	GTO 02	
135	XEQ 05	*Compute numerator of R_2*
136	RCL 06	*Compute denominator of R_2*
137	XEQ 06	
138	÷	
139	STO 02	*Store computed R_2*
140	RTN	
141♦	LBL 02	*Store R_2 and invalidate V_{PL} and V_{PH}*
142	STO 02	
143	SF 01	*Must recompute V_{PL} and V_{PH} for user-defined R_2*
144	RTN	
145♦	LBL D	*Store or compute R_3*
146	FS?C 22	
147	GTO 03	
148	XEQ 05	*Compute numerator of R_3 (same as R_2)*
149	RCL 06	*Compute denominator of R_3*
150	RCL 05	
151	−	
152	÷	
153	RCL 11	
154	RCL 12	
155	−	
156	÷	
157	STO 03	*Store computed R_3*
158	RTN	
159♦	LBL 03	*Store R_3 and invalidate V_{PL}h and V_{PH}*
160	STO 03	
161	SF 01	*Must recompute V_{PL} and V_{PH} for user-defined R_3*
162	RTN	
163♦	LBL E	*Store or compute R_4*

164	FS?C 22	
165	GTO 04	
166	XEQ 07	*Get denominator (also used for computing f or duty cycle)*
167	RCL 00	
168	×	
169	RCL 13	
170	×	
171	1/x	
172♦	LBL 04	*Store entered or computed* R_4
173	STO 04	
174	RTN	
175♦	LBL 05	*Compute numerator common to* R_2 *and* R_3
176	XEQ 01	*Recompute* V_{PL} *and* V_{PH} *if necessary*
177	RCL 05	
178	XEQ 06	
179	RCL 01	
180	×	
181	CHS	
182	RTN	
183♦	LBL 06	*Compute* $(V_{OL} - V_{PL} - V_{OH} + V_{PH}) * X + V_{OH} * V_{PL} - V_{PH} * V_{OL}$
184	RCL 07	
185	RCL 11	
186	−	
187	RCL 08	
188	−	
189	RCL 12	
190	+	
191	×	*Multiply* $(V_{OL} - V_{PL} - V_{OH} + V_{PH})$ *by* V_{POS} *or* V_{NEG} *(now in Y)*
192	RCL 08	
193	RCL 11	
194	×	
195	+	
196	RCL 12	
197	RCL 07	
198	×	
199	−	
200	RTN	

Registers and Flags

Register	Use
00	*Frequency (Hz)*
01,02,03,04	*Resistors* R_1, R_2, R_3, *and* R_4 *(Ohms)*
05,06	V_{NEG} *and* V_{POS} *(Volts)*
07,08	V_{OL} *and* V_{OH} *(Volts)*
11,12	V_{PL} *and* V_{PH} *(Volts)*

13	*Capacitor C_1 (Farads)*
09,14,15,10	*Temporary registers*

Flag	Meaning
01	*V_{PL} and V_{PH} need to be recomputed*
22	*User supplied input*

Revision History

2009-May-25 — Initial release of HP-41C/CV/CX version.

2009-Jun-11 — Fixed a bug where the data entry flag sometimes wasn't cleared even though there had been no data entry.

2015-Jun-23 — Fixed a bug in the subroutine for computing the denominator of the duty cycle, wherein V_{PL} and V_{PH} were accidentally interchanged. Bug was in web-site listing only, not in original program.

Like One person likes this.

Related Articles

If you've found this article useful, you may also be interested in:

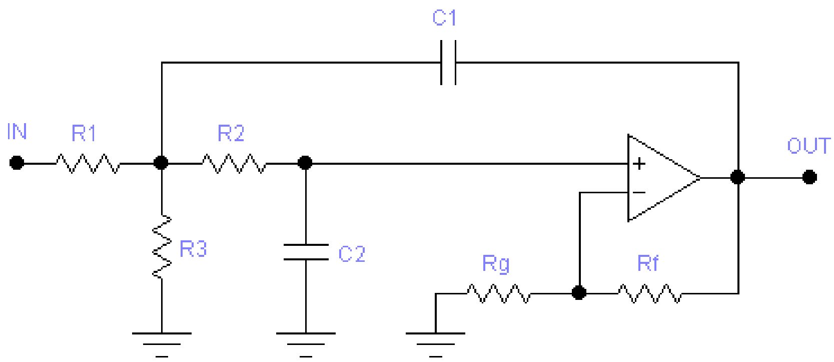
- *Op-Amp Gain and Offset Design with the HP-41C Programmable Calculator*
- *Low-Sensitivity Sallen-Key Filter Design with the HP-41C Programmable Calculator*
- *Low-Sensitivity Sallen-Key Filter Design with the HP-67 Programmable Calculator*
- *Op-Amp Gain and Offset Design with the HP-67 Programmable Calculator*
- *Op-Amp Oscillator Design with the HP-67 Programmable Calculator*
- *Resistor Network Solver for the HP-67 Programmable Calculator*
- *A Matrix Multi-Tool for the HP 35s Programmable Calculator*
- *Curve Fitting for the HP 35s Programmable Calculator*

Low-Sensitivity Sallen-Key Filter Design with the HP-41C Programmable Calculator

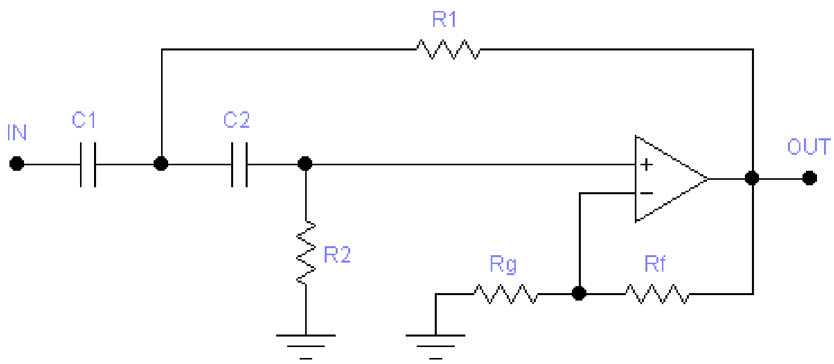
December 21, 2008

This program, which I originally wrote for the HP-67 calculator, addresses the problem of designing second order single op-amp low- and high-pass filters using the Sallen-Key topology.

The Sallen-Key filter topology has the advantage of using a minimum of components. The simplest Sallen-Key filters use only two resistors and two capacitors. Additional resistors may be added for input attenuation (low-pass only) and gain adjustment. The following schematics illustrate these generalized Sallen-Key circuits:



Generalized Sallen-Key low-pass filter



Generalized Sallen-Key high-pass filter

The equations governing the low-pass filter are as follows,

$$f = \frac{1}{2\pi\sqrt{\frac{R_1 R_2 R_3 C_1 C_2}{R_1 + R_3}}}$$

$$Q = \frac{1}{2\pi f\left(R_2 C_2 + \frac{R_1 R_3 C_2}{R_1 + R_3} - \frac{R_1 R_3 C_1 R_F}{(R_1 + R_3) R_G}\right)}$$

$$H = \frac{R_3\left(1 + \frac{R_F}{R_G}\right)}{R_1 + R_3}$$

where f is the filter's cut-off frequency, Q is its "quality", and H is its gain at the cut-off frequency. The corresponding equations for the high-pass filter are:

$$f = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}}$$

$$Q = \frac{1}{2\pi f\left(R_1 C_1 + R_1 C_2 - \frac{R_2 C_2 R_F}{R_G}\right)}$$

$$H = 1 + \frac{R_F}{R_G}$$

The mathematically inclined will notice that in each case, there are three equations in either nine or ten variables. Thus there is no single "right" solution. At least six or seven of the variables have to be decided arbitrarily (f, Q, H, and three or four component values), at which point the remaining variables (component values) can be solved for. An article by Texas Instruments suggests a number of simplifications to help one choose component values, but this just adds the complication of which simplification to choose.

I recently came across a pair of application notes by National Semiconductor which gives a procedure for designing Sallen-Key filters to minimize the effect of component value tolerances on the performance of the filter. A side-effect of this procedure is to reduce the number of inputs to five: f, Q, H, R_F, and R, where the latter is simply an indication of the magnitude of resistor values desired for R_1, R_2, and R_3. The procedure then dictates how all the other values are chosen, even adjusting for available "real-world" values part way through the solution.

The program presented here implements this procedure, with some minor changes:

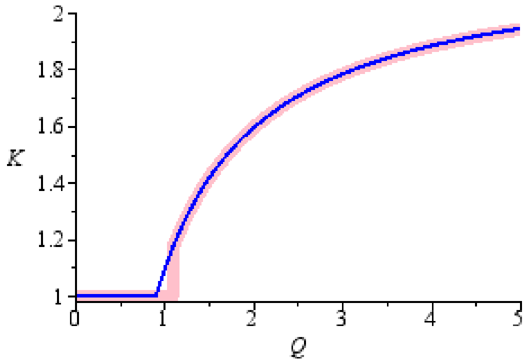
1. Instead of asking for a desired resistor magnitude, R, the program asks for a capacitor magnitude, C, since in my experience, the capacitors drive the design.

2. The formula given for internal gain variable K in the procedure (please refer to the application note) seems to have been derived empirically, and has a jump at $Q = 1.1$.

To make the formula simpler to implement, I modified it slightly to:

$$K = \max\left(\frac{2.2\,Q - 0.9}{Q + 0.2},\ 1\right)$$

The graphs of the original (pink) and revised (blue) formulae show the difference. Testing has shown that the resulting solution is generally at most one real-world capacitor value increment different (with corresponding changes in resistor values of course).



Also, if $H > K$, then K is set equal to H, since otherwise it will not be possible to achieve the desired gain.

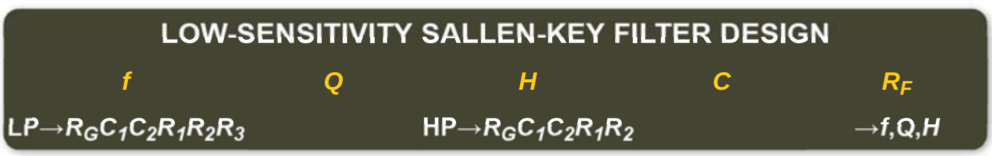
For the low-pass circuit, the desired gain, H, is achieved by a combination of input attenuation, α (controlled by R_1 and R_3), and the internal gain, K, of the filter (controlled by R_F and R_G). The division of this gain between the two stages depends on H and Q and is chosen to minimize the sensitivity of the circuit to component tolerances. For example, for $H = 1$ and $Q = 2$, the attenuator gain is $\alpha = 0.629$ and the internal gain is $K = 1.59$, for a net gain of $H = \alpha K = 1$.

The high-pass circuit has no attenuation stage, so H must be at least 1, and higher for $Q > 0.917$. If too low a value is entered for H, it is increased as necessary to make the circuit solvable.

For either circuit, to achieve a result with minimum component sensitivity without regard to gain, set $H = 0$. The program will automatically choose the optimal value for K (and thus H). The resulting gain will be output when determining the performance.

Using the Program

First type in the program and save it, or read it from a previously recorded magnetic card. The card should be labelled as follows:



Filter Design from Specifications

Example: Design a 500Hz low-pass unity-gain filter with a Q of 2, using capacitors in the 10nF range, and a 47kΩ resistor for R_F:

Description	Keystrokes	Display
Use engineering notation	ENG 2	0.00 00
Enter f	500 a	500. 00
Enter Q	2 b	2.00 00
Enter H	1 c	1.00 00
Enter C	10 EEx CHS 9 d	10.0 -09
Enter R_F	47 EEx 3 e	47.0 03
Compute R_G	A	79.5 03
Enter real-world R_G *and compute* C_1	82 EEx 3 R/S	31.6 -09
Enter real-world C_1 *and compute* C_2	33 EEx CHS 9 R/S	3.16 -09
Enter real-world C_2 *and compute* R_1	3.3 EEx CHS 9 R/S	15.4 03
Enter real-world R_1 *and compute* R_2	15 EEx 3 R/S	95.9 03
Enter real-world R_2 *and compute* R_3	100 EEx 3 R/S	26.1 03
Enter real-world R_3	27 EEx 3 R/S	27.0 03

Notes

If a resistor is to be omitted (open circuit), this program displays a value of zero for the resistance. This is different than some of my other programs, which display a “large” value representing infinity.

When the value for R_G is displayed as zero, meaning it can be omitted, the value of R_F will not matter any more, and R_F can be replaced by a direct connection.

Filter Performance from Chosen Components

During the calculation of the solution above, we’ve entered real-world values in response to each computed value. The real-world values of C_1 and C_2 are used when computing the values of R_1, R_2, and R_3. However, the real-world values of each of those resistors does not affect the computed value of the remaining ones. Thus, the final filter may not perform exactly as specified. To find out how it does perform, follow these steps:

Description	Keystrokes	Display
Compute resulting f	E	491. 00
Compute resulting Q	R/S	1.86 00
Compute resulting H	R/S	1.01 00

A High-Pass Example

Using the parameters already entered for the low-pass filter above, determine the components for a high-pass filter:

Description	Keystrokes	Display
Compute R_G for high-pass filter	C	79.5 03
Enter real-world R_G and compute C_1	82 EEx 3 R/S	31.4 -09
Enter real-world C_1 and compute C_2	33 EEx CHS 9 R/S	3.18 -09
Enter real-world C_2 and compute R_1	3.3 EEx CHS 9 R/S	9.59 03
Enter real-world R_1 and compute R_2	10 EEx 3 R/S	97.0 03
Enter real-world R_2	100 EEx 3 R/S	0.00 00

Now determine the predicted actual performance:

Description	Keystrokes	Display
Compute resulting f	E	482. 00
Compute resulting Q	R/S	1.96 00
Compute resulting H	R/S	1.57 00

Notice that H is higher than the specified unity gain. This is because the filter is not possible to construct with unity gain when $Q = 2$. The smallest possible gain is $H = 1.59$ (which due to real-world components, has become $Q = 1.89$ and $H = 1.57$). To achieve $H = 1$, you will need either a pre-attenuator with low output impedance, or a post-attenuator with high input impedance.

Program Listing

Line	Instruction	Comments
01♦	LBL "SK"	
02♦	LBL a	*Enter and store f*
03	STO 11	
04	RTN	
05♦	LBL b	*Enter and store Q*
06	STO 12	
07	RTN	
08♦	LBL c	*Enter and store H (gain)*

09	STO 13	
10	RTN	
11♦	LBL d	*Enter and store C (capacitor scale)*
12	STO 00	
13	RTN	
14♦	LBL e	*Enter and store* R_F
15	STO 04	
16	RTN	
17♦	LBL A	*Low-pass filter:* $R_G, C_1, C_2, R_1, R_2, R_3$
18	CF 00	
19	GTO 00	
20♦	LBL C	*High-pass filter:* R_G, C_1, C_2, R_1, R_2
21	SF 00	
22♦	LBL 00	*Forward solution*
23	RCL 12	
24	2.2	
25	×	
26	.9	
27	–	
28	RCL 12	
29	.2	
30	+	
31	÷	*(2.2Q-0.9)/(Q+0.2)*
32	1	
33	x≤y?	
34	x↔y	
35	STO 15	*K = max(1,(2.2Q-0.9)/(Q+0.2))*
36	RCL 13	
37	x>y?	
38	STO 15	*K = max(H,1,(2.2Q-0.9)/(Q+0.2))*
39	RCL 15	
40	÷	*H/K*
41	x=0?	
42	1	*Use α = 1 if H/K = 0 (because H was 0)*
43	FS? 00	
44	1	*Always use α = 1 for a high-pass filter*
45	STO 14	*α = H/K (always 1 for a high-pass filter or H = 0)*
46	RCL 04	
47	RCL 15	
48	1	
49	–	
50	x≠0?	
51	÷	$R_G = R_F/(K-1)$ *if K ≠ 1, or zero if K = 1*
52	R/S	*Display* R_G *and let user change it*
53	STO 05	
54	.1	*Initialize n to √0.1;*

55	√x	
56	XEQ 07	$n(1+\sqrt{(1+4Q^2(1+n^2)(K-1))})/(2Q(1+n^2))$
57	RCL 08	*$\sqrt{0.1}$ was stored here by subroutine 7*
58	x≤y?	
59	x↔y	$max(n, n(1+\sqrt{(1+4Q^2(1+n^2)(K-1))})/(2Q(1+n^2)))$
60	FS? 00	*High-pass filter?*
61	XEQ 06	$2nQ/(1+\sqrt{(1+4Q^2(K-1-n^2))})$
62	STO 08	
63	RCL 00	
64	RCL 08	
65	÷	$C_1 = C/n$
66	R/S	*Display C_1 and let user change it*
67	STO 06	
68	RCL 08	
69	RCL 00	
70	×	$C_2 = nC$
71	R/S	*Display C_2 and let user change it*
72	STO 07	
73	RCL 06	
74	×	
75	√x	$\sqrt{(C_1C_2)}$
76	XEQ 04	*Multiply by $2\pi f$ and take reciprocal*
77	STO 09	$N = 1/(2\pi f\sqrt{(C_1C_2)})$
78	RCL 07	
79	RCL 06	
80	÷	
81	√x	$n = \sqrt{(C_2/C_1)}$
82	XEQ 03	$2nQ/(1+\sqrt{(1+4Q^2(K-1-n^2))})$ *or* $n(1+\sqrt{(1+4Q^2(1+n^2)(K-1))})/(2Q(1+n^2))$
83	STO 08	
84	RCL 09	
85	×	
86	STO 03	$(R_1\|\|R_3) = nN$
87	RCL 14	
88	÷	$R_1 = (R_1\|\|R_3)/\alpha$
89	R/S	*Display R_1 and let user change it*
90	STO 01	
91	RCL 09	
92	RCL 08	
93	÷	$R_2 = N/n$
94	R/S	*Display R_2 and let user change it*
95	STO 02	
96	RCL 03	
97	1	
98	RCL 14	
99	–	$(1-\alpha (R_1\|\|R_3))$
100	x≠0?	

101	÷	*$R_3 = (R_1 \| \| R_3)/(1-\alpha)$ if $\alpha \neq 1$, or zero otherwise*
102	R/S	*Display R_3 and let user change it*
103	STO 03	
104	RTN	
105♦	LBL 04	*Multiply by $2\pi f$ and take reciprocal*
106	RCL 11	*f*
107	×	
108♦	LBL 01	*Multiply by 2π and take reciprocal*
109	2	
110	×	
111	π	
112	×	
113	1/x	
114	RTN	
115♦	LBL 03	*Compute either $2xQ/(1+\sqrt{(1+4Q^2(K-1-x^2))})$ or $x(1+\sqrt{(1+4Q^2(1+x^2)(K-1))})/(2Q(1+x^2))$*
116	FS? 00	*High-pass filter?*
117	GTO 07	
118♦	LBL 06	*Subroutine to compute $2xQ/(1+\sqrt{(1+4Q^2(K-1-x^2))})$*
119	STO 08	*Save x for later use*
120	RCL 12	
121	2	
122	×	*(2Q x)*
123	×	*(2xQ)*
124	LASTx	*(2Q 2xQ)*
125	x^2	*($4Q^2$ 2xQ)*
126	RCL 15	
127	1	
128	–	
129	RCL 08	
130	x^2	
131	–	*($K-1-x^2$ $4Q^2$ 2xQ)*
132	×	*($4Q^2(K-1-x^2)$ 2xQ)*
133	XEQ 09	*($1+\sqrt{(1+4Q^2(K-1-x^2))}$ 2xQ)*
134	÷	
135	RTN	
136♦	LBL 07	*Subroutine to compute $x(1+\sqrt{(1+4Q^2(1+x^2)(K-1))})/(2Q(1+x^2))$*
137	STO 08	*Save x in register 8 for later use both by this subroutine and the caller*
138	XEQ 08	*(2Q $1+x^2$)*
139	x^2	
140	×	*($4Q^2(1+x^2)$)*
141	RCL 15	
142	1	
143	–	*(K-1 $4Q^2(1+x^2)$)*
144	×	*($4Q^2(1+x^2)(K-1)$)*
145	XEQ 09	*($1+\sqrt{(1+4Q^2(1+x^2)(K-1))}$)*

146	RCL 08	
147	×	*($x(1+\sqrt{}(1+4Q^2(1+x^2)(K\text{-}1)))$)*
148	RCL 08	
149	XEQ 08	*($2Q$ $1+x^2$ $x(1+\sqrt{}(1+4Q^2(1+x^2)(K\text{-}1)))$)*
150	×	*($2Q(1+x^2)$ $x(1+\sqrt{}(1+4Q^2(1+x^2)(K\text{-}1)))$)*
151	÷	
152	RTN	
153♦	LBL 08	*Subroutine to populate stack with $2Q$ $1+x^2$*
154	x^2	
155	1	
156	+	
157	RCL 12	
158	2	
159	×	
160	RTN	
161♦	LBL 09	*Subroutine to compute $1+\sqrt{}(1+x)$*
162	1	
163	+	
164	√x	
165	1	
166	+	
167	RTN	
168♦	LBL E	*Compute actual f, Q, and Gain*
169	XEQ 02	*R_1 or $(R_1\|\|R_3)$*
170	RCL 06	
171	×	
172	STO 08	*Save R_1C_1 for use in calculating Q*
173	RCL 02	
174	RCL 07	
175	×	
176	STO 09	*Save R_2C_2 for use in calculating Q*
177	×	
178	√x	
179	XEQ 01	*Multiply by 2π and take reciprocal*
180	STO 10	*Save actual f for use in calculating Q*
181	R/S	*Display actual f*
182	RCL 09	*(R_2C_2)*
183	RCL 08	*(R_1C_1 R_2C_2)*
184	FS? 00	*High-pass filter?*
185	x⇔y	*(R_2C_2 R_1C_1)*
186	1	
187	RCL 15	
188	−	
189	×	
190	+	
191	XEQ 02	*R_1 or $(R_1\|\|R_3)$*

192	`RCL 07`	
193	`×`	
194	`+`	
195	`RCL 10`	*Recall actual frequency*
196	`×`	
197	`XEQ 01`	*Multiply by 2π and take reciprocal*
198	`R/S`	*Display actual Q*
199	`RCL 04`	
200	`RCL 05`	
201	`x≠0?`	
202	`÷`	
203	`1`	
204	`+`	
205	`XEQ 02`	R_1 *or* $(R_1\|\|R_3)$
206	`×`	
207	`RCL 01`	
208	`÷`	
209	`RTN`	*Return actual Gain*
210♦	`LBL 02`	*Return either* R_1 *or* $(R_1\|\|R_3)$
211	`RCL 01`	
212	`FS? 00`	*High-pass filter?*
213	`RTN`	*Return just* R_1
214	`1/x`	
215	`RCL 03`	
216	`x≠0?`	R_3 *exists? (0 means not)*
217	`1/x`	
218	`+`	
219	`1/x`	
220	`RTN`	

Registers and Flags

Register	Use
00	*C – capacitor scale*
01,02,03	R_1, R_2, R_3 *– filter resistors*
04,05	R_F, R_G *– feedback resistors*
06,07	C_1, C_2 *– capacitors*
08	*n – variable used during computation*
09	*N – variable used during computation*
11	*f – cutoff frequency*
12	*Q – filter quality*
13	*H – overall gain*
14	*α – input attenuator gain*
15	*K – internal gain*
10	*Temporary register*

Flag	Meaning
00	*High-pass filter*

Revision History

2008-Dec-21 — Initial release.

References

1. *Analysis of the Sallen-Key Architecture (Rev.B)*, Texas Instruments Application Report SLOA024B, James Karki, 2002

2. *Low-Sensitivity, Lowpass Filter Design*, National Semiconductor (now Texas Instruments) Application Note OA-27, Kumen Blake, 1996

3. *Low-Sensitivity, Highpass Filter Design*, National Semiconductor (now Texas Instruments) Application Note OA-29, Kumen Blake, 1996

Like Be the first of your friends to like this.

Related Articles

If you've found this article useful, you may also be interested in:

- *Op-Amp Gain and Offset Design with the HP-41C Programmable Calculator*
- *Op-Amp Oscillator Design with the HP-41C Programmable Calculator*
- *Low-Sensitivity Sallen-Key Filter Design with the HP-67 Programmable Calculator*
- *Op-Amp Gain and Offset Design with the HP-67 Programmable Calculator*
- *Op-Amp Oscillator Design with the HP-67 Programmable Calculator*
- *Resistor Network Solver for the HP-67 Programmable Calculator*
- *A Matrix Multi-Tool for the HP 35s Programmable Calculator*
- *Curve Fitting for the HP 35s Programmable Calculator*