USING YOUR HP-41 ADVANTAGE:
Electrical Circuits for Students

Here is THE program (designed by engineering graduates who wish THEY had had this when they were in school!) to help the engineering student sail through his/her electrical fundamentals courses.

This program (and your HP-41CV or CX, with the HP-41 Advantage Module) is all you may ever need to calculate voltages, currents and powers in typical AC circuits!

The program allows for sinusoidal current or voltage sources (V or I), variable frequency (f or w), and passive impedance elements of resistance (R), inductance (L), and capacitance (C).

With a plain HP-41CV, you can "crunch" a circuit with up to 4 nodes; with maximum Extended Memory added, you can tackle up to 12 nodes.

PLUS, you can model mutual inductance, solve for input impedances and Thevenin's and Norton's equivalents, perform parallel-series conversions, Y-Delta conversions, and much more!

ALL THIS, with program listings, bar code, and quick-and-easy instructions for getting started.

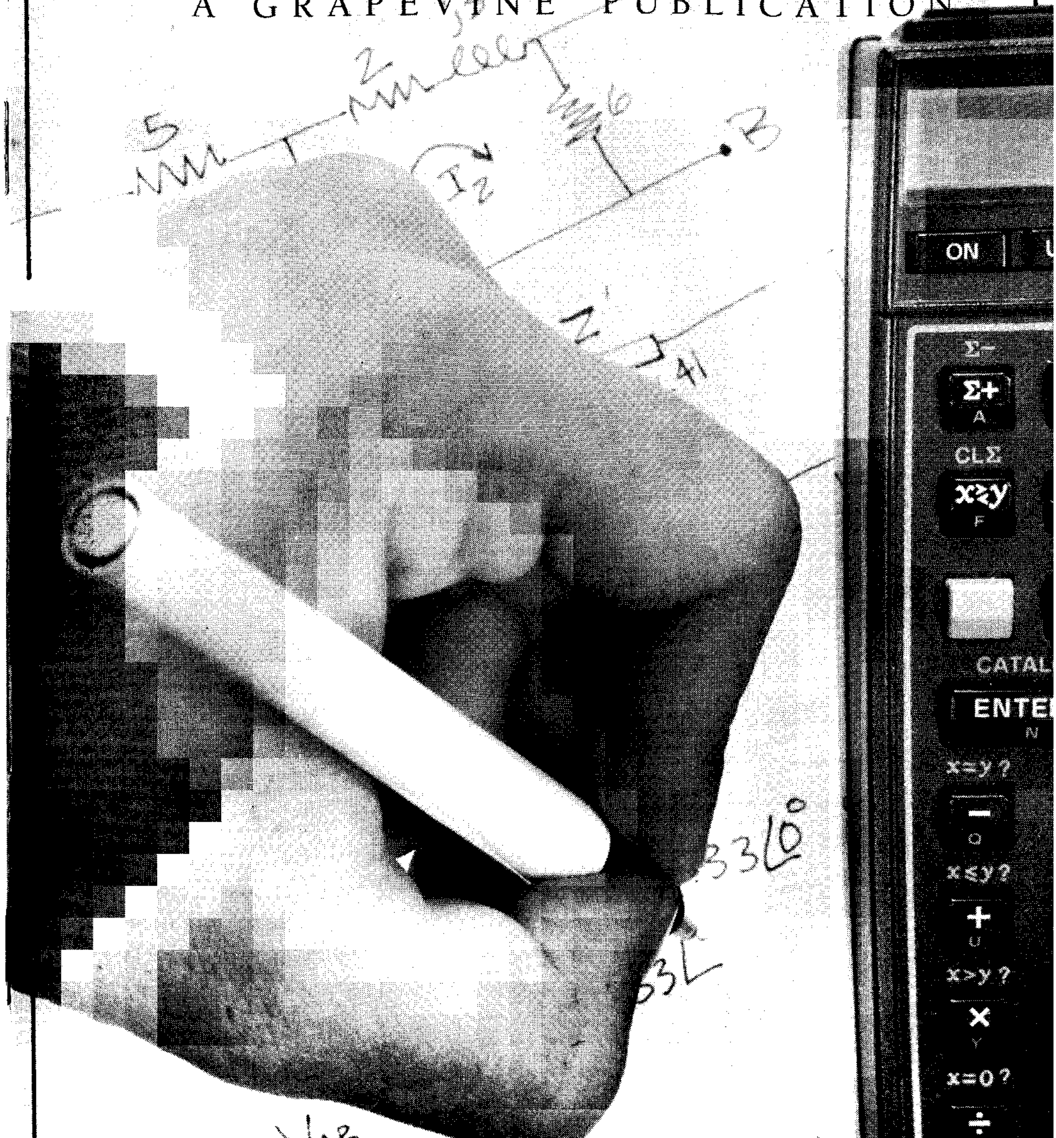
92235S

ISBN 0-931011-03-5

Using Your HP-41

ADVANTAGE

A GRAPEVINE PUBLICATION



Electrical Circuits for Students

USING YOUR HP-41 ADVANTAGE:

ELECTRICAL CIRCUITS FOR STUDENTS

by Chris Coffin and Ted Wadman
Illustrated by Robert L. Bloch
Produced by Gregg Kleiner, Carol Sweet, and Soraya Simons

Grapevine Publications, Inc.
P.O. Box 118
Corvallis, Oregon 97339-0118 USA

For the sake of brevity, the terms "Advantage," "Advantage Module," or "HP-41 Advantage" have been used herein to denote "The HP-41 Advantage Advanced Solutions Pac," which is the proper and reserved name for Hewlett-Packard's plug-in module and its instruction manual for the HP-41 handheld computer system. We extend our thanks to Hewlett-Packard Company for producing such top-quality products and documentation.

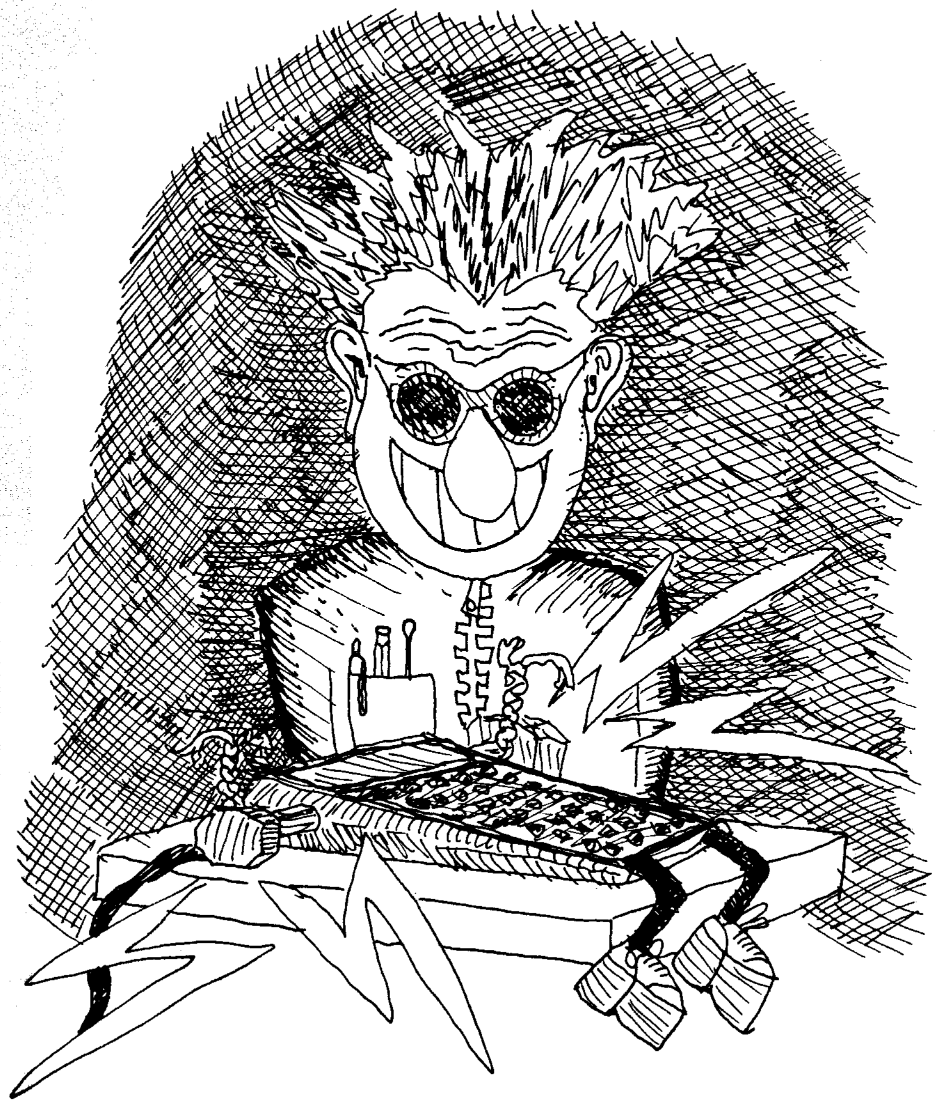
ACKNOWLEDGEMENTS

Equations and formulas used in this book and its program may be found in "Schaum's Outline of Theory and Problems of Electric Circuits," by Joseph A. Edminister, M.S.E. © 1965 McGraw-Hill, Inc., New York.

Special thanks and salutes are due to Chris Bunsen, who conceived and developed the HP-41 Advantage, and who encouraged the development of this book.

Cover photo by Tom Brennan.

Printed in the United States of America
ISBN 0-931011-03-5



LOADING THE PROGRAM

YOUR CALCULATOR'S CONFIGURATION

In order to use this program, you'll need (at least):

- an HP-41CV or HP-41CX;
- the HP-41 Advantage module;
- this book;
- just a little time to read through this, load the program, and practice with it.

(You probably knew all of that, but we needed to warm up here a little bit.)

Also, if you have an Extended Functions module (and/or one or two Extended Memory modules), you'll find those useful, so keep them plugged into your calculator.

Now, assuming that you have everything on this checklist, the first step is to load the program.

Starting on page 8, you'll find a program listing and a set of barcode.

But before you get to that...

A word from our sponsor:

ARE YOU RUSTY ?

In this book, you'll need to be able to follow keystroke procedures, load, pack and execute a program, etc. But we really don't want to take the room here to go over all of that (besides, you didn't want this book to weigh over 3 pounds, did you?). So we're going to assume you know these things. For example, do you know:

- How to read keystroke notation? (e.g. [XEQ] [ALPHA] SIZE [ALPHA] 228)
- How USER mode works?
- How to select FIX, ENG or SCI notation?
- How the Stack works, including [+],[−],[×],[÷],[X◄►Y],[R↓], [STO] and [RCL]?
- How to SIZE and PACK your calculator's memory?
- How to read and key in program steps such as:

```
XROM "C+"
ARCL 19
X<> 00
DSE IND Z
"⊢:A B  C D"
FC?C 03
```

- How to move around in--and edit--a program (in case you mis-key some step)?

If these things are new to you (or have faded into the last module of your brain's Extended Memory), you'll have a LOT of difficulty in continuing here until you take a slight detour for a refresher course. We recommend one of the following:

- Look up in your Owner's handbook whatever is on this checklist that you don't remember;

- Read "An Easy Course in Programming the HP-41," by Chris Coffin and Ted Wadman (see page 73 for details).

(Our favorite suggestion is the second one.)

... OK, if you're all refreshed and reminded, it's time to load the program.

PREPARING YOUR CALCULATOR'S MEMORY

The "CV" version uses exactly 190 program registers; the "CX" version uses exactly 188 registers;

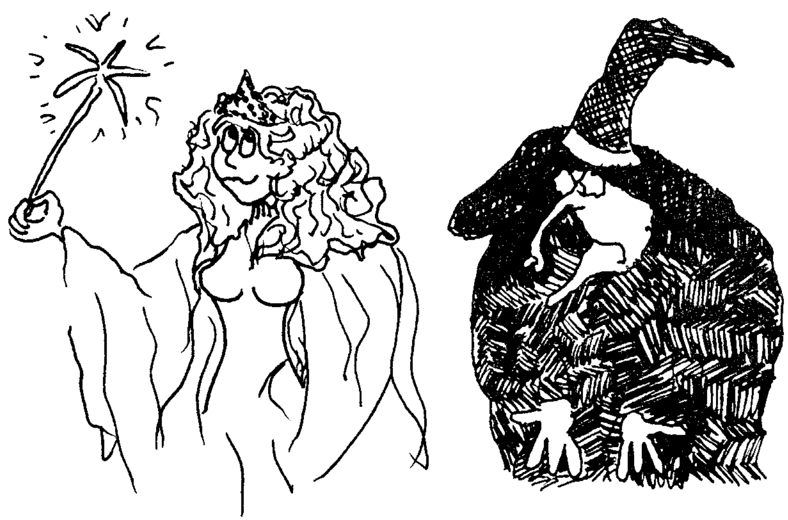
On top of that, you will need anywhere from 37 to 129 data registers to analyze a circuit.

So... make room for the program in your calculator! (Do that now)

Once you have made sufficient space:

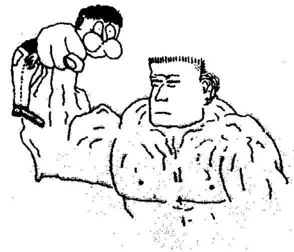
IF you have an HP-41 Wand, you can connect that now, turn to the barcode, and read it in.

If you *DON'T* have a wand... we have some good news and some bad news...



The good news is: Very shortly, you will have totally mastered all aspects of keying in an HP-41 program.

The bad news is: This is because you have about 780 lines of code to key in (yes, Virginia–by hand.)



Now... before you hit the ceiling, consider this:

If you have an HP-41 Card Reader or another such storage device, the first thing you'll be doing (and we're pretty sure about this)–is to make yourself a copy (and a back-up copy, preferably) of the correctly loaded program. For heavens sake, do so, but run it a few times first, so that you're sure it's correct and compiled (i.e. up to maximum speed).

If you don't have a Card Reader, but you do have some Extended Memory, you can store the program there when you're not using it (but when you ARE using it, you'll need to copy it into the calculator and–depending on the size of the circuit you want to crunch–clear it from Extended Memory to make room for your circuit data).

Admittedly, this isn't strictly convenient, but if you don't want to keep the program loaded into main memory all the time, this beats the heck out of the only other alternative.

In either case, when you prepare program memory, right before you start keying it in, you'll see

"00 REG nnn"

in the display. MAKE A NOTE of that number nnn. Then...start poking them keys!


```
01♦LBL "Z"
02 1
03 STO 07
04 STO 08
05 2
06 STO 09
07 XEQ 25
08♦LBL "R"
09♦LBL 00
10 DEG
11 SF 00
12 CF 01
13 CF 02
14 CF 03
15 CF 04
16 CF 05
17 CF 06
18 SF 16
19 CF 21
20 FS? 55
21 SF 21
22 SF 27
23 CF 29
24 .9
25 STO 20
26 "Aa b  C"
27 PROMPT
28♦LBL J
29 GTO 00
30♦LBL A
31 22
32 XEQ 28
33 GTO 13
34♦LBL C
35 ISG 20
36 GTO 26
37♦LBL b
38 SF 03
39♦LBL a
40 "N"
41 RCL 08
42 XEQ 24
43 STO 08
44 "E"
45 RCL 09
46 XEQ 24
47 RCL 09
48 X>Y?
49 CLX
50 STO 04
51 RDN
52 STO 09
53 XEQ 25
54 RCL 04
55 1
56 +
57 FS?C 03
58 CLX
59♦LBL 13
60 SF 25
61 MSIJ
62 FC?C 25
63 GTO 00
64 ISG 20
65 RCL 07
66 2
67 PI
68 *
69 STO 00
70 /
71 "F"
72 XEQ 24
73 RCL 00
74 *
75 STO 07
76♦LBL 15
77 FIX 2
78 SF 06
79 MRR+
80 STO 00
81 MRR+
82 MRR-
83 X<>Y
84 XEQ 19
85 STO 01
86 RDN
87 STO 02
88 XEQ 35
89 X<>Y
90 1 E2
91 STO 00
92 RDN
93 4
94 X>Y?
95 X<>Y
96 RDN
97 6
98 X<=Y?
99 X<>Y
100 RDN
101 X<>Y
102 RCL 01
103 RCL 02
104 X<> Z
105 XEQ IND T
106 "⊦?"
107 SF 02
108 CF 22
109 AVIEW
110 PSE
111♦LBL 01
112♦LBL 16
113 VIEW X
114 STOP
115 FC? 22
116 GTO 14
117 "Z I  V"
118 PROMPT
119 GTO 01
120♦LBL A
121♦LBL 04
122 "Z"
123 4
124 GTO 13
125♦LBL B
126♦LBL 05
127 "I"
128 5
129 GTO 13
130♦LBL C
131♦LBL 06
132 "V"
```

```
133 6
134♦LBL 13
135 RDN
136 FS?C 06
137 RTN
138♦LBL 14
139 R↑
140 ST* 00
141 RDN
142 SIGN
143 ST* 00
144 CLX
145 LASTX
146 ST+ 00
147 J-
148 RDN
149 XEQ 18
150 RCL 00
151 MSR+
152 RDN
153 MSR+
154 RDN
155 MSR+
156 FC? 10
157 GTO 15
158 GTO J
159♦LBL E
160 ISG 20
161♦LBL 03
162♦LBL 17
163 VIEW X
164 FS?C 07
165 STOP
166 "PS,DY CX   */"
167 SF 04
168 CF 02
169 SF 07
170 PROMPT
171 GTO 03
172♦LBL B
173 SF 03
174♦LBL b
175 GTO 23
176♦LBL F
177 SF 03
178 X<>Y
179♦LBL G
180 SF 07
181 FC?C 02
182 SF 02
183 FC?C 03
184 RDN
185 GTO 07
186♦LBL J
187 DSE 20
188 AOFF
189 CF 04
190 GTO IND 20
191♦LBL e
192 SF 03
193♦LBL E
194 XEQ 18
195 FS?C 03
196 XROM "CINV"
197 XROM "C*"
198 XEQ 19
199 GTO 03
200♦LBL C
201 XEQ 20
202 GTO 03
203♦LBL c
204 XEQ 21
205 GTO 03
206♦LBL A
207 XEQ 18
208 XROM "C+"
209 XEQ 19
210 GTO 03
211♦LBL a
212 XEQ 18
213 XROM "CINV"
214 RDN
215 RDN
216 XROM "CINV"
217 XROM "C+"
218 XROM "CINV"
219 XEQ 19
220 GTO 03
221♦LBL D
222 SF 00
223 FC?C 01
224 SF 01
225♦LBL d
226 CF 07
227 FC?C 00
228 SF 00
229 FS? 00
230 CF 01
231♦LBL 07
232 FS? 04
233 GTO 17
234 GTO 16
235♦LBL H
236 SF 03
237 "H"
238♦LBL I
239 SF 07
240 FC? 03
241 "I"
242 "⊢:A B  C D"
243 ABS
244 RDN
245 17
246 STO 21
247 CLX
248 2
249 PROMPT
250 CF 03
251 GTO 07
252♦LBL A
253 ST- 21
254♦LBL B
255 ST- 21
256♦LBL C
257 ST- 21
258♦LBL D
259 RDN
260 LASTX
261 FC?C 03
262 GTO 13
263 X<>Y
264 STO IND 21
265 DSE 21
266 X<>Y
267 STO IND 21
268 GTO 07
269♦LBL 13
270 RCL IND 21
271 DSE 21
272 RCL IND 21
273 GTO 07
274♦LBL 18
275 FC? 00
276 GTO 13
277 P-R
278 RDN
279 RDN
280 P-R
281 RDN
282 RDN
283 XEQ 21
284♦LBL 13
285 FS? 01
286 XEQ 21
287 FS? 04
288 XEQ 20
289 RTN
290♦LBL 19
291 FS? 04
292 XEQ 21
293 FS? 01
294 XEQ 20
295 FC? 00
296 RTN
297 XEQ 20
298 R-P
299 RDN
300 RDN
301 R-P
302 RDN
303 RDN
304 RTN
305♦LBL 20
306 STO 21
307 RDN
308 RCL 07
309 *
310 X<0?
311 1/X
312 R↑
313 RCL 07
314 *
315 X<0?
316 1/X
317 GTO 13
318♦LBL 21
319 STO 21
320 RDN
321 XEQ 22
322 R↑
323 XEQ 22
324♦LBL 13
325 STO T
326 CLX
327 RCL 21
328 RTN
329♦LBL 22
330 X<0?
331 1/X
332 RCL 07
333 X=0?
334 X<>Y
335 /
336 RTN
337♦LBL 23
338 RCL 15
339 RCL 14
340 XEQ 18
341 STO 14
342 RDN
343 STO 15
344 RCL 13
345 RCL 12
346 RCL 11
347 RCL 10
348 XEQ 18
349 STO 16
350 RDN
351 STO 17
352 RDN
353 STO 12
354 RDN
355 STO 13
356 RDN
357 FC?C 03
358 GTO 13
359 XROM "C+"
360 RCL 15
361 RCL 14
362 XROM "C+"
363 XROM "CINV'
364 RCL 13
365 RCL 12
366 XROM "C*"
367 RCL 17
368 RCL 16
369 XROM "C*"
370 GTO 14
371♦LBL 13
372 XROM "C*"
373 RCL 15
374 RCL 14
375 XROM "C/"
376 RCL 17
377 RCL 16
378 XROM "C+"
379 RCL 13
380 RCL 12
381 XROM "C+"
382♦LBL 14
383 STO 10
384 X<>Y
385 STO 11
386 X<>Y
387 RCL 13
388 RCL 12
389 XROM "C/"
390 RCL 15
391 RCL 14
392 XROM "C*"
393 XEQ 19
394 STO 12
395 RDN
396 STO 13
```

```
397 RCL 15
398 RCL 14
399 RCL 11
400 RCL 10
401 XROM "C*"
402 RCL 17
403 RCL 16
404 XROM "C/"
405 RCL 11
406 RCL 10
407 XEQ 19
408 STO 10
409 RDN
410 STO 11
411 RDN
412 STO 14
413 RDN
414 STO 15
415 GTO 17
416♦LBL 24
417 "⊦="
418 ARCL X
419 "⊦?"
420 PROMPT
421 ABS
422 RTN
423♦LBL 25
424 FIX 0
425 RCL 09
426 3
427 *
428 23
429 +
430 STO 19
431 1
432 +
433 RCL 08
434 2
435 *
436 +
437 STO 18
438 LASTX
439 .1
440 %
441 +
442 "R"
443 ARCL 18
444 MATDIM
445 INT
446 "R"
447 ARCL 19
448 MATDIM
449 "R22"
450 RCL 09
451 3 E-3
452 +
453 MATDIM
454 RTN
455♦LBL 26
456 SF 05
457 ISG 20
458 RCL 19
459 XEQ 34
460 RCL 18
461 XEQ 34
462♦LBL 27
463 22
464 XEQ 28
465 STO 01
466 STO 02
467 STO 03
468 STO 04
469 STO 05
470 STO 06
471 FS?C 05
472 MSIJ
473♦LBL 08
474 MRR+
475 X<0?
476 SF 03
477 ABS
478 STO 00
479 MRR+
480 MRR+
481 X<>Y
482 XEQ 20
483 XEQ 35
484 RDN
485 5
486 X<=Y?
487 GTO 13
488 RDN
489 2
490 GTO 09
491♦LBL 13
492 X=Y?
493 GTO 14
494 RDN
495 4
496 GTO 09
497♦LBL 14
498 RDN
499 6
500♦LBL 09
501 RDN
502 RDN
503 X<>Y
504 ST+ IND Z
505 DSE Z
506 X<>Y
507 ST+ IND Z
508 FS?C 03
509 GTO 08
510 RCL 02
511 RCL 01
512 SF 04
513 RCL 06
514 RCL 05
515 +
516 X≠0?
517 CLST
518 RDN
519 SF 25
520 XROM "CINV"
521 CF 25
522 STO 01
523 X<>Y
524 STO 02
525 X<>Y
526 FS?C 02
527 RTN
528 RCL 04
529 RCL 03
530 XROM "C*"
531 ST+ 05
532 X<>Y
533 ST+ 06
534 RCL 19
535 XEQ 28
536 SF 03
537♦LBL 10
538 RCL 06
539 CHS
540 STO 06
541 RCL 05
542 CHS
543 STO 05
544 XEQ 32
545 X=0?
546 GTO 13
547 RDN
548 MRR-
549 MR
550 XROM "C+"
551 MSR+
552 X<>Y
553 MS
554♦LBL 13
555 XEQ 33
556 FS?C 03
557 GTO 10
558 RCL 18
559 XEQ 28
560 XEQ 30
561 XEQ 33
562 XEQ 30
563 -1
564 ST* 01
565 ST* 02
566 FS?C 02
567 GTO 13
568 XEQ 31
569 XEQ 33
570 XEQ 31
571♦LBL 13
572 22
573 XEQ 28
574 RDN
575 CF 04
576 DIM?
577 X>Y?
578 GTO 27
579 RCL 18
580 XEQ 28
581 "⊦,R"
582 ARCL 19
583 MSYS
584 "R22"
585 MSIJA
586♦LBL 02
587 SF 04
588 "P  I VG"
589 PROMPT
590 GTO 02
591♦LBL 28
592 "R"
593 FIX 0
594 ARCL X
595 MRIJA
596 0
597 RTN
598♦LBL c
599 STO 00
600 XEQ 11
601 RDN
602 RDN
603 XROM "C/"
604 GTO 13
605♦LBL B
606 XEQ 29
607 GTO 13
608♦LBL A
609 XEQ 29
610 RCL 04
611 RCL 03
612 XROM "C*"
613 GTO 13
614♦LBL C
615 STO 00
616 XEQ 12
617♦LBL 13
618 FIX 2
619 XEQ 19
620 SF 07
621 GTO E
622♦LBL 11
623 STO 00
624 RCL 19
625 XEQ 28
626 XEQ 32
627 X=0?
628 GTO 13
629 MRR-
630 MR
631♦LBL 13
632 XEQ 33
633 RDN
634 XEQ 32
635 X=0?
636 ENTER↑
637 X=0?
638 RTN
639 RDN
640 MRR-
641 MR
642 RTN
643♦LBL 29
644 STO 21
645 22
646 XEQ 28
647 SF 02
648 MR
649 ABS
650 STO 00
651 XEQ 35
652 RCL 21
653 X=Y?
654 XEQ 27
655 0
656 FS? 10
657 MSIJ
658 FC?C 02
659 GTO 14
660 I+
```

```
661 FS? 09
662 MSIJ
663 RCL 21
664 GTO 29
665♦LBL 14
666 XEQ 12
667 RCL 04
668 RCL 03
669 XROM "C+"
670 STO 03
671 X<>Y
672 STO 04
673 X<>Y
674 RCL 02
675 RCL 01
676 XROM "C*"
677 RCL 06
678 RCL 05
679 XROM "C+"
680 RTN
681♦LBL 12
682 RCL 00
683 XEQ 11
684 XROM "C-"
685 RTN
686♦LBL 30
687 RCL 00
688 STO 21
689 INT
690 X=0?
691 SF 02
692 X=0?
693 SF 03
694 1
695 %
696 +
697 STO 00
698 FC?C 03
699 XEQ 31
700 RCL 21
701 STO 00
702 RTN
703♦LBL 31
704 XEQ 32
705 RCL 00
706 FRC
707 .2
708 *
709 +
710 MSIJ
711 RCL 01
712 MR
713 +
714 MS
715 J-
716 RCL 02
717 MR
718 +
719 MS
720 I-
721 RCL 01
722 MR
723 +
724 MSR+
725 RCL 02
726 CHS
727 MR
728 +
729 MS
730 RTN
731♦LBL 32
732 RCL 00
733 INT
734 2
735 *
736 MSIJ
737 RTN
738♦LBL 33
739 RCL 00
740 FRC
741 1 E2
742 *
743 X<> 00
744 INT
745 1 E2
746 /
747 RCL 00
748 +
749 STO 00
750 RTN
751♦LBL 34
752 "R"
753 FIX 0
754 ARCL X
755 DIM?
756 0
757 MSIJ
758 MS
759 MATDIM
760 X<>Y
761 MATDIM
762 RTN
763♦LBL 35
764 RCL 00
765 1 E2
766 /
767 STO 00
768 LASTX
769 X<>Y
770 FRC
771 *
772 RCL 00
773 INT
774 ABS
775 X<>Y
776 STO 00
777 END
```

ROW 1: LINES 1-7

ROW 2: LINES 8-13

ROW 3: LINES 14-20

ROW 4: LINES 20-26

ROW 5: LINES 26-30

ROW 6: LINES 31-36

ROW 7: LINES 36-42

ROW 8: LINES 43-52

ROW 9: LINES 53-61

ROW 10: LINES 61-70

ROW 11: LINES 71-78

ROW 12: LINES 78-85

ROW 13: LINES 86-94

ROW 14: LINES 95-105

ROW 15: LINES 106-113

ROW 16: LINES 113-117

ROW 17: LINES 118-125

ROW 18: LINES 126-134

ROW 19: LINES 135-144

ROW 20: LINES 145-152

ROW 21: LINES 153-158

ROW 22: LINES 159-166

ROW 23: LINES 166-166

ROW 24: LINES 167-173

ROW 25: LINES 174-180

ROW 26: LINES 180-187

ROW 27: LINES 187-194

ROW 28: LINES 194-199

ROW 29: LINES 200-205

ROW 30: LINES 205-210

ROW 31: LINES 211-217

ROW 32: LINES 218-223

ROW 33: LINES 224-230

ROW 34: LINES 230-236

ROW 35: LINES 236-242

ROW 36: LINES 242-245

ROW 37: LINES 246-253

ROW 38: LINES 254-261

ROW 39: LINES 261-268

ROW 40: LINES 269-275

ROW 41: LINES 276-285

ROW 42: LINES 285-291

ROW 43: LINES 291-297

ROW 44: LINES 297-306

ROW 45: LINES 307-318

ROW 46: LINES 318-325

ROW 47: LINES 325-335

ROW 48: LINES 336-345

ROW 49: LINES 346-354

ROW 50: LINES 355-363

ROW 51: LINES 363-370

ROW 52: LINES 371-378

ROW 53: LINES 379-389

ROW 54: LINES 390-399

ROW 55: LINES 400-407

ROW 56: LINES 407-416

ROW 57: LINES 417-423

ROW 58: LINES 424-433

ROW 59: LINES 434-443

ROW 60: LINES 443-449

ROW 61: LINES 449-456

ROW 62: LINES 456-461

ROW 63: LINES 462-470

ROW 64: LINES 471-479

ROW 65: LINES 479-486

ROW 66: LINES 487-496

ROW 67: LINES 497-506

ROW 68: LINES 508-515

ROW 69: LINES 517-525

ROW 70: LINES 527-534

ROW 71: LINES 535-543

ROW 72: LINES 545-550

ROW 73: LINES 552-557

ROW 74: LINES 559-562

ROW 75: LINES 563-568

ROW 76: LINES 569-573

ROW 77: LINES 575-580

ROW 78: LINES 581-584

ROW 79: LINES 586-588

ROW 80: LINES 590-595

ROW 81: LINES 597-604

ROW 82: LINES 605-609

ROW 83: LINES 611-617

ROW 84: LINES 619-624

ROW 85: LINES 625-630

ROW 86: LINES 631-638

ROW 87: LINES 640-646

ROW 88: LINES 647-652

ROW 89: LINES 654-659

ROW 90: LINES 661-666

ROW 91: LINES 667-676

ROW 92: LINES 677-684

ROW 93: LINES 686-693

ROW 94: LINES 695-702

ROW 95: LINES 704-710

ROW 96: LINES 712-719

ROW 97: LINES 720-727

ROW 98: LINES 729-737

ROW 99: LINES 739-745

ROW 100: LINES 746-754

ROW 101: LINES 755-761

ROW 102: LINES 763-771

ROW 103: LINES 773-777

PROOFREADING YOUR PROGRAM

With the program now keyed into your calculator, press [SHIFT] [GTO[[.] [.] to pack the program and make sure there's an END to it.

While in PRGM mode, you should now see:

"00 REG nnn,"

where nnn is a number that differs from what you noted there before you started. The difference should be 190. If this checks, it doesn't necessarily mean that the program is correctly loaded, but it's a good omen, and you should proceed. ——→

— — — — — — — — — —

If that difference is NOT correct, you have miskeyed some step, in which case you should go to the beginning of the program and check it, step by step, against the listing here in the book (better yet, if you have a printer, get a printout of the program as you now have it, and compare THAT against the book).

Chances are, you have either skipped or duplicated some step. Another common error is neglecting to use the [XEQ] key properly. If you're guilty of this heinous crime, the evidence will be unmistakable: A program line which appears here WITHOUT quotation marks (" ") will nonetheless appear WITH the little superscript T in the display (OR vice versa). This is a no-no! The written quotes and that T in the display should EXACTLY correspond.

Also, program labels (LBL's) will appear in the printed listing with ◄►'s in front of them, but no such ◄►'s will appear in the display.

But these are the only differences between the printed listing and the way things should look in your display. Everything else should match EXACTLY. Please–don't go on from this point until you can arrive at the correct reading for

"00 REG nnn"

If you are working with a plain HP-41CV, you may now skip ahead. ——→

But if you have any Extended Memory capability with your calculator, you may want to make the following editing adjustments to your program. These adjustments will allow you to "crunch" much larger circuits.

Be sure to follow these keystrokes exactly (and be patient with your keystrokes: with a program of this size, it takes awhile for the calculator to do each editing task!):

[SHIFT] [GTO] [.] [.]
[SHIFT] [GTO] [ALPHA] Z [ALPHA]

Next, go into PRGM mode: [PRGM] (if you are not already there). Then:

[SHIFT] [GTO] [.] 443
[◄–] [◄–]
[ALPHA] RY [ALPHA] (this is now step 442)

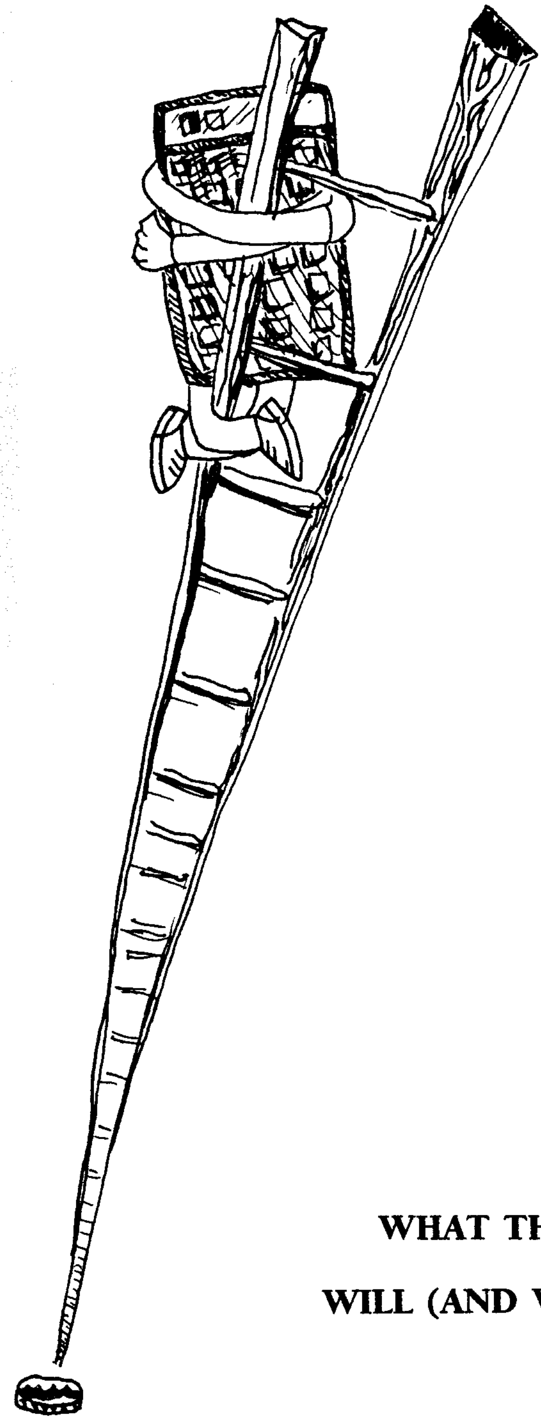
[SHIFT] [GTO] [.] 438
[◄–] [◄–] [◄–]

[SHIFT] [GTO] [.] 432
[◄–] [◄–]
[ALPHA] Y
[SHIFT] [ASTO] 18 [ALPHA] (this is now step 432)

[SHIFT] [GTO] [.] 017
[◄–] [◄–] [◄–] [◄–] [◄–] [◄–] [◄–] (seven of 'em)
1
[XEQ] [ALPHA] [X◄►F] [ALPHA] (this is now step 12)

[PRGM]
[SHIFT] [GTO] [.] [.]

You now have the "CX" version of the program!



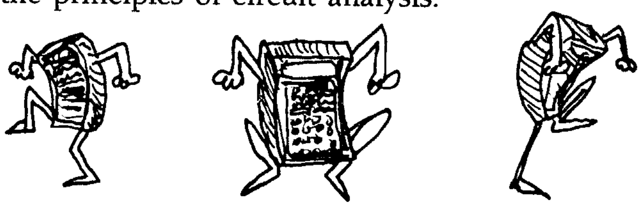
WHAT THIS PROGRAM WILL (AND WON'T) DO

THIS PROGRAM WILL:

- Solve for complex voltage, current, power and gain, anywhere in a sinusoidal AC circuit (see page 34 for a table showing the size limits of the circuit).
- Allow easy storage and recall of these and other complex calculations, including parallel-to-series and Y-to-Delta conversions, complex arithmetic, and polar-rectangular equivalents.
- Allow you to save, expand or alter the circuit you analyze, including the frequency.
- Accept circuit elements V and I (complex sources) and impedances R, L, C, either in units of ohms, henrys and farads, or combined as complex ohms (either in polar or rectangular form).

THIS PROGRAM WILL NOT:

- Solve for any impedence value in the circuit, nor solve any type of circuit other than sinusoidal AC;
- Detect resonance or do multiple-frequency sweeps;
- Detect bad input, mixed units, mixed notation, or otherwise do your common-sense thinking for you. A calculator is a great number-cruncher, but it's a lousy engineer, and it certainly won't take the place of your studying and understanding the principles of circuit analysis.



(If this is all the information you need to get started, skip ahead to page 29, If not, keep reading here.)

SOLVING CIRCUITS

Set your machine aside for a few minutes and review a few basics:

What's a circuit?

An electrical circuit is just that–a closed loop of conductive material, around which current will flow, from higher potential (i.e. voltage) to lower.

In many respects, electrical flow is like a cyclical flow of water--like a pump that sends water up a hill to flow down, encountering rocks or water wheels which dissipate--or utilize--the energy of the flow.

You can think of the impedances placed in an electrical circuit as those rocks or water wheels; and you can think of the voltage or current sources to be like the pumps that send the water up the hill to complete the cycle.

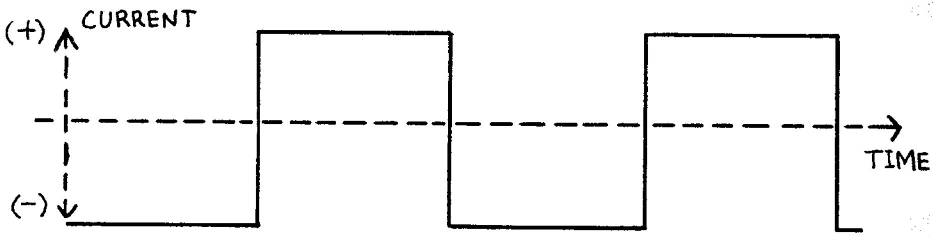
The analogy is even more basic than that: Electrical current is the collective motion of many charges, just as the flowing stream is the collective motion of many water droplets. And voltage is like the height from which the water flows down: the longer the total fall, the more potential energy each particle of the falling material has.

This means that a voltage source is like a water pump whose flow rate may vary, but which always pumps water up to the same height on the hill; a current source would then be a pump that always pumps the same volume of water per minute, but it may not always deliver it to the same height on the hill. Get the picture?

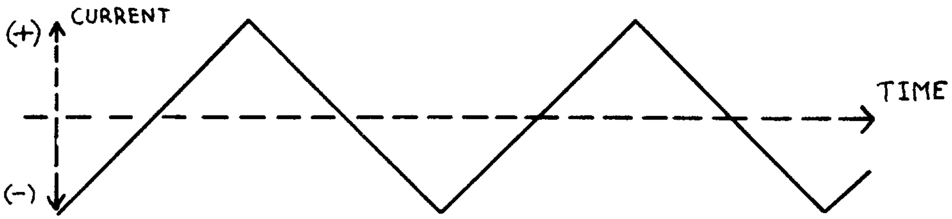
WHAT KINDS OF CIRCUITS CAN THE PROGRAM "CRUNCH?"

Of course, our analogy falls apart when we start talking about Alternating Current (AC), where the direction of the "hill" and the "waterflow" is periodically (and quite rapidly) reversing itself. The rate of this periodic reversal is called the frequency (F).

There are many different possible patterns to this flip-flopping of direction. It might look like this:

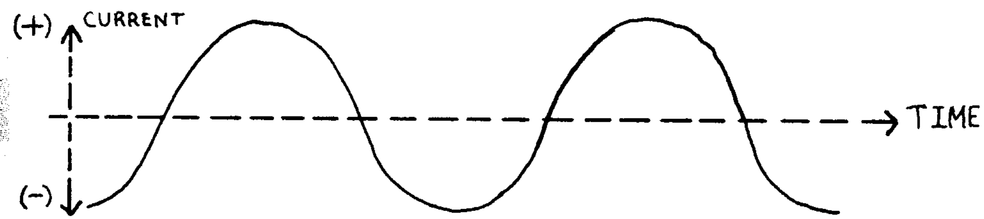


Or this:



But by far the most common is the following pattern, called SINUSOIDAL AC, because it is a simple sine or cosine pattern.

This is what this program is all about. It will NOT handle any kind of Alternating Current except this:



Notice that as the frequency gets smaller, the whole pattern would flatten out and begin to look more and more like our simple, steady waterfall analogy. In the limit, if the frequency approached zero, we would indeed have our waterfall--called Direct Current, since it wouldn't reverse directions at all.

WHAT CIRCUIT ELEMENTS CAN YOU USE?

Sources:

$V = V_{real} + j(V_{imag.})$, in volts;

$I = I_{real} + j(I_{imag.})$, in amperes;

Impedances:

R = resistance, in ohms;

L = inductance, in henrys;

C = capacitance, in farads;

general reactive impedance, $X = wL - 1/wC$, in ohms;

general complex impedance, $Z = R + jX$, in ohms.

All complex numbers may be represented in any of three forms:

$X + jY$	(rectangular form)
$Z \angle a°$	(polar or "phasor" form)
R and (either) L or C	(physical form)

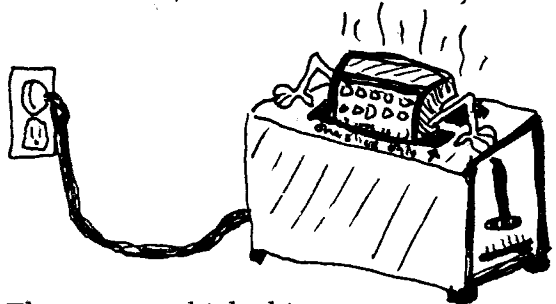
(If this is all clear and straightforward to you, go on ahead to page 34)

Need some more details? Another quick review...

Circuit elements are simply the sources and impedances you place in the closed loop of wire we call a circuit. The sources you'll use in this program are Voltage (V) and Current (I) sources. They are the "pumps" in the cycle.

There are three types of impedances to electrical flow (like rocks in the stream bed) that you can use in this program: Resistance (R), Capacitance (C), and Inductance (L).

We're all familiar with resistance, measured in units of ohms. It's what happens when your electric toaster heats up: Current (I) flows from high voltage (V) to ground voltage (0)–just like water coming down the hill to sea level. But strategically placed wires–with a high resistance (R)–present an electrical barrier to this flowing current. This barrier forces the circuit to expend some energy in overcoming it. This energy comes out in the form of heat, and thus browneth your Roman Meal.



The rate at which this energy comes out is called the power (P). The equations which relate all this are:

$$V = IR$$

$$P = IV = V^2/R = RI^2$$

Simple enough.

But resistance is not the whole story. There are other kinds of impedance which rear their (not always ugly) heads, but only when current is changing–either in magnitude or (as with AC) in direction. These forms of impedance are called reactive impedances, because they arise as "reactions" to the changing current.

Remember how that works? When a current is changing, the flowing charges are either changing direction or stopping–in either case, accelerating. And we all recall, of course, that an accelerating charge produces a magnetic field. But a magnetic field, in turn, exerts a force on ANY charge that happens to be MOVING within that field.

It turns out, therefore, that the net effect of a changing current is that it produces a magnetic field which in turn exerts its OWN force on that current, setting up a kind of counter-current. And the directions work out so that this counter-current tends to oppose (i.e. impede) the current that was changing in the first place!

Well, L and C accomplish this impeding in different ways, and so they have characteristically different effects on the current they are impeding. For the purpose of this program, which is only for sinusoidal voltages and currents, the magnitude of the REACTIVE impedance, X, from an inductor, L, is:

$$X = wL$$

where $w = 2\pi F$ (remember, F is the frequency with which the current is alternating its direction).

By comparison, the REACTIVE impedance, X, from a capacitor, C, is:

$$X = -1/wC$$

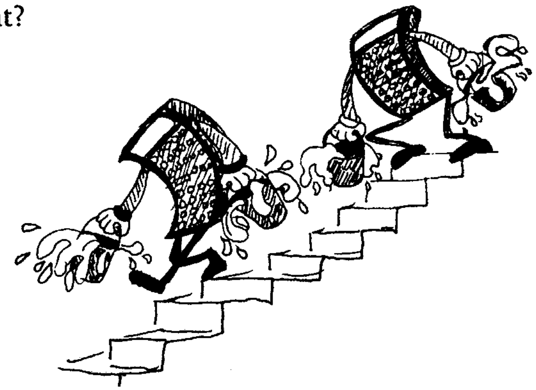
Notice that this impedance is a negative number. Just what does that mean? Do you get negative heat from it when you put a capacitor in your toaster (frozen muffins?...)

Not really. The Resistance portion and the Reactance portion of an impedance are combined numerically as a complex number--with a real portion (the Resistance) and an imaginary portion (Reactance).

The resistance is always the tangible heat or work-producing portion of the impedance. But that word "imaginary" doesn't mean that the reactance is just something we've dreamed up out of thin air. It really does exist. When we use an imaginary number to describe something physical, we are merely referring to the way it behaves: it may or may not be tangible, depending upon how it combines with other complex quantities. After all, the basis for an imaginary number,

$$j = \sqrt{-1}$$

becomes a real number when you multiply it by itself (or some multiple of itself), right?



So there you have it: An impedance (Z) is a complex quantity:

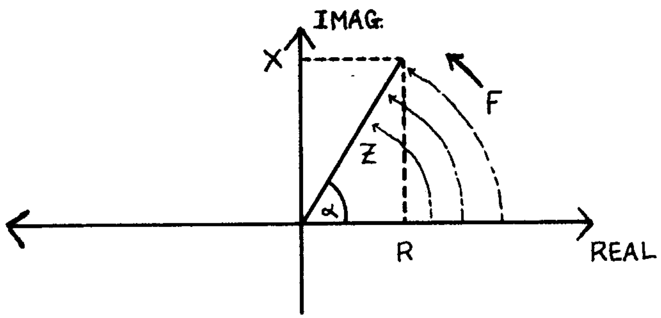
$$Z = R + jX$$

You can forget about your toaster equations. The more generally correct forms are:

$$V = IZ$$

$$P = IV = V^2/Z = ZI^2$$

And ALL these numbers are complex–not just the Z: When you have sinusoidally alternating currents and voltages, they too can be represented by complex numbers that are rotating about the complex number plane–at a frequency of F Hertz:



How do we know this? Well, when you measure voltage, current or power with meters, you are measuring only the real portions of those quantities; if you trace the behavior of ONLY the real portion (i.e. the x-component) of a complex number as it rotates around the axis, it WILL describe a sinusoidal pattern (Aha)!

Notice that this rotating complex number can be described in rectangular coordinates as R (resistance) and X (reactance); or in polar coordinates as Z (magnitude) and α (phase angle).

Either of these forms is acceptable to this program. And in the case of an impedance, there is a third possibility: just knowing the frequency, F, and the R and (L or C) values in a complex impedance, it's easy to convert to the picture above–and the program does that for you, if you prefer to use that physical language rather than the ohms the machine has to use for its mathematics.

Now go back to page 29 and see if that summary doesn't make more sense to you.

CIRCUIT SIZE LIMITS

With this memory config., a circuit with this number of nodes can have up to this many elements

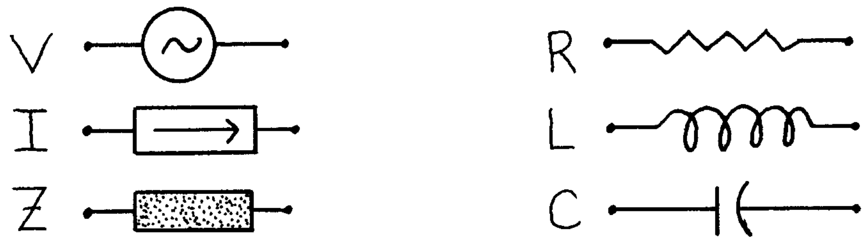
	1	2	3	4	5	6	7	8	9	10	11	12
A plain HP-41CV	32	27	20	10								
An HP-41CX or Equiv. (i.e.. HP-41CV w/X-Func.)	34	33	33	32	31							
An HP-X-CX or Equiv. plus 1 Ext. Mem. Mod.	34	33	33	32	31	31	30	29	29			
An HP-41CX or Equiv. plus 2 Ext. Mem. Mod.	34	33	33	32	31	31	30	29	29	28	27	27

If this is all you need to know to get started, go right on ahead to page 36.

Not so fast? One more review...

CIRCUIT DIAGRAM NOTATION

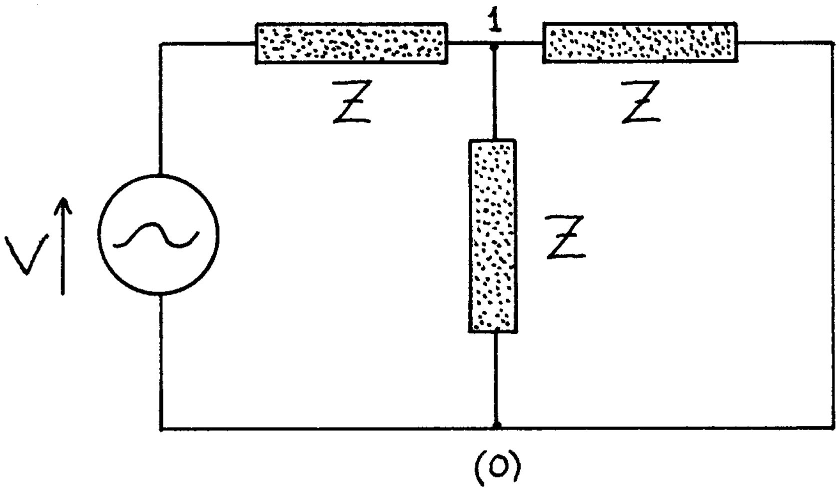
You already know what elements we're talking about here: V, I, R, L, C, and Z. Here's how we represent them as being connected into a circuit:



Keep in mind that you can always combine an R and either an L or C into a Z. This saves time and represents just one element in the circuit, rather than two (and this saves memory for the calculator).

But what's a node, anyway?

Here is a typical circuit that you might be analyzing:



Here you see a sinusoidal voltage source and three impedances. The point labelled "1" in the circuit is called a node, because it is the junction of three (3) or more elements. In this program, you can name other points as nodes also, but you HAVE to name all points that connect three or more elements.

Notice that the ground point (i.e. point of zero voltage) is also a node, but by convention, it is always labelled point "0" and not counted when you're counting the number of nodes in the circuit. The program recognizes this convention, so when you say "Node O," it understands that you mean the ground point.

As far as the program is concerned, then, the above circuit has one (1) node.

USING THE PROGRAM

COLD-STARTING THE PROGRAM

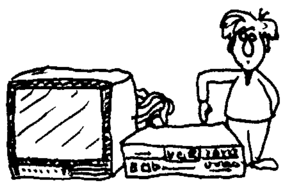
No matter what you're doing with your calculator, as long as this program is loaded into main memory, you can start with

[XEQ] "Z" ([XEQ] [ALPHA] Z [ALPHA])

This "Zeros out" your circuit data–a clean slate.

Try it right now.

NOTE: You'll get a NONEXISTENT message if you don't have the SIZE set to at least 037 (or 032 for the CX version). If this happens, just adjust your SIZE and start the program again.



WARM-STARTING THE PROGRAM

Another way to start the program is to use

[XEQ] "R"

This will bring you to the same place as [XEQ] "Z", but it won't erase any circuit data that may have been in the calculator before you started the program. Remember that circuit data are just numbers in registers, and those values will stay there even when you're not using this program–unless you change them or clear them.

One side note at this point: If you have a printer and would like to see how this program operates with a printer, hook it up to your HP-41, set it to NORMAL mode (if there is such a setting on the printer), and start again with one of the above procedures.

ROAD MAP: You Are Here

OK, where's "here?" Where does this program "go" when you tell it to do something?

The best way to picture how this program "moves" around is to keep your place here but flip over to the inside of the back cover of this book.---

That's the road map.

You are now at the Main Menu. See where that is on the map?

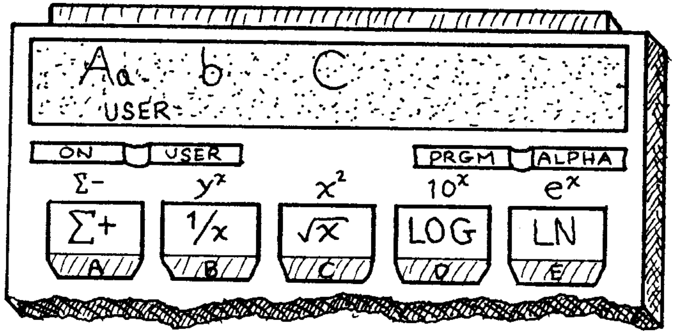
The program is really just four different menus that you choose for your various calculations. So if you're already comfortable with the idea of a menu–and with using the top two rows of keys on your calculator as the selectors for these menus–then skip right now to page 40. ⟶

MENUS

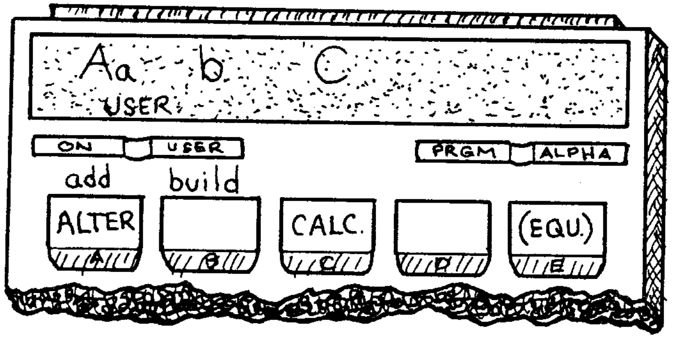
As you may know, with USER mode turned on, the top two rows of keys provide up to 15 different one-key program selectors. In the first row, you can select keys A-E or a-e (the shifted versions of each key). In the second row, you have keys F-J.

Now, a program menu is simply a place in the program where the calculator stops to let you select one of these special keys. And to help you remember what each of the keys will do, a few reminder symbols appear in the display, each symbol directly (more or less) above the key it represents.

For example, at the Main Menu (now appearing in the display), this:



means this:



You simply press the key of the selection you want, and off you go. Be sure to remember that whenever there are two characters over a key, the first one stands for the key itself, and the second one stands for the SHIFTED version of that key.

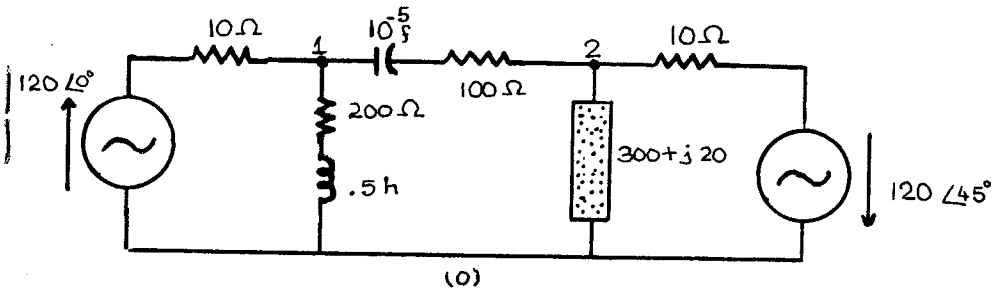
add	build				SHIFTED FIRST ROW OF KEYS
ALTER		CALCU-LATE		(EQUA-TIONS)	FIRST ROW OF KEYS
				(JUMP)	SECOND ROW OF KEYS

Note: If you're ever confused as to which menu you're in or what your options are, you can usually find your way to a menu simply by pressing [R/S]

One more note: In each menu you encounter, there will be some possible selections that NEVER show up in the display. These are the HIDDEN MENU (the keys appearing in parentheses on the Road Map); you'll be learning about these very shortly. But DON'T press a menu key that does not appear at all on the Road Map.

BUILDING A CIRCUIT

Time now to hit the Road. Suppose, just for laughs that you had the following 60-Hz circuit:



And suppose that, for more laughs, you wanted to calculate the voltages at node 1 and node 2, and the currents through the (300 + j20) impedance and the .5–henry inductor.

First of all, how would you use the program to build a picture of the circuit for the calculator to solve?

SOLUTION SUMMARY

The first thing you need to consider is how much memory this circuit will require for storage and crunching:

If N is the number of Nodes, and E is the number of elements, then your minimum SIZE is

$4N^2 + 2N + 3E + 25$ for the CV version

and $2N + 3E + 24$ for the CX version

You will also need at least $4N^2 + 1$ registers of Extended Memory to run your circuit with the CX version.

Thus, for this example circuit, N = 2, and E = 7. So CV SIZE should be at least 066; CX SIZE should be at least 049, with at least 17 available registers of Extended Memory.

Now, how would you draw the picture for the calculator? In 25 words or less, this is how:

YOU PRESS		YOU SEE
[XEQ] "Z"		"Aa b C"
[shift] [b]		"N = 1?"
2 [R/S]		"E = 2?"
7 [R/S]		"F = ?"
60 [R/S]		"Z?" (pause) 0.00
0	[ENTER]	
120	[ENTER]	
−0.01	[R/S]	"Z I V"
[C]		
		"Z?" (pause) 0.00
0	[ENTER]	
10	[ENTER]	
0.01	[R/S]	"Z I V"
[A]		
		"Z?" (pause) 0.00
[shift]	[d]	0.00
.5	[ENTER]	
200	[ENTER]	
1	[R/S]	"Z I V"
[A]		
		"Z?" (pause) 0.00
−1 E−5	[ENTER]	
100	[ENTER]	
1.02	[R/S]	"Z I V"
[A]		
		"Z?" (pause) 0.00
[D]		0.00
20	[ENTER]	
300	[ENTER]	
2	[R/S]	"Z I V"
[A]		
		"Z?" (pause) 0.00
0	[ENTER]	
10	[ENTER]	
−2	[R/S]	"Z I V"
[A]		
		"Z?" (pause) 0.00
[shift]	[d]	0.00
45	[ENTER]	
120	[ENTER]	
2	[R/S]	"Z I V"
[C]		
		"Aa b C"

Now, if that seemed trivial and completely self-explanatory, go on ahead to page 50. But keep reading here if you want another look at the reasons for those keystrokes....

First: [XEQ] "Z" (if you haven't already)

When the Main Menu appears, press the [b] key (the [shift][B] key) as the display indicates. As you can see on the road map, this selection will let you "build" a picture of a circuit for the calculator.

You see: "N=1?"

You're being asked for tne number of nodes in your circuit. In this circuit there are two nodes, so you would press

2 [R/S]

If there had been only one node in your circuit, you could have pressed 1 [R/S] OR just [R/S], and the calculator would have kept its current value of 1 node. This is the way the program usually works: when the calculator prompts you for some value(s), it will have something "suggested" already. If you accept its suggestion, just press [R/S]. If not, just key in the correct value(s) and THEN press [R/S].

Next, you see: "E=2?"

Although there are 9 actual physical elements in the circuit, remember that a pure resistance and a pure capacitor or inductor can be combined to form a general complex impedance, Z. Doing this wherever possible will save you and your calculator time and memory.

Therefore, you press 7 [R/S]

Next you see: "F = (something)?" Press: 60 [R/S], because the frequency in this problem is 60 Hz.

ELEMENT VALUES IN THE STACK

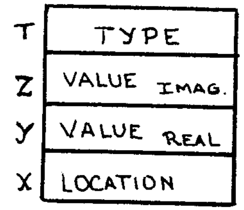
Now you see:

"Z?" (pause) 0.00 (or some other number)

This is the start of the element checking/changing routine. Again, the calculator is asking you something, offering suggestions, and then waiting for your response. Here's what it's looking for:

The momentary "Z?" tells you that it "thinks" this element is an impedance (i.e. not a V or I source).

The value that appears next is just one of four values now in the calculator's stack. Here's what the stack looks like at the moment:



[X◄►Y] AND [R↓]

Of course, you can't see all this in the display at once, so you do what you would normally do to review the stack–use [R↓].

Go ahead and press [R↓] once. What happens?

The [R↓] key, along with every other key on the second row, is part of the HIDDEN MENU; that is, it never shows on the display's version of the menu, but it will do what it's told, nevertheless.

Press [R↓] three more times, to get back to where you started.

Now press [X◄►Y] (twice). It, too, works normally.

But notice that both [R↓] and [X◄►Y] keep turning on and off the little 2 (flag 2) in the display. This is to tell you something:

Whenever flag 2 is OFF, this means that you're currently viewing the REAL (first) part of a (two-part) complex number. And whenever flag 2 is ON, you're looking at the IMAGINARY (second) part of the number.

Yet when you're back looking at the location value, the 2 IS on. Why? This is so that when you roll-down once to look at the real part of the complex number, the 2 will go off, and you're back "in step." So get used to the idea that when the suggested values appear at this point, you'll be looking at the location value, and this is signified by the 2.

This is all good and fine, but exactly what ARE those numbers in the stack? Look at them one at a time:

THE LOCATION VALUE

The first number (it's in the X-register) is the LOCATION of the element in the circuit. This number is in From.To format. That is, an element from node 1 to node 2 has a location value of 1.02. From node 2 to ground is 2.00 (or just plain 2 is fine). From ground to node 1 is 0.01.

Notice that each part has two digits–always!

One other important thing about the location value: you also use this value to indicate whether the element is one of several elements strung together in a series in this particular branch of the circuit. You should indicate this with a negative sign: Whenever this element is NOT the LAST one in a particular branch, use the negative sign.

So the voltage source in our circuit has a location of −0.01, because there is also a 10-ohm resistor in that branch. And that resistor would have a location of +0.01, because it IS the last element in the branch. Notice that when we say "last," that means we have chosen a "direction" of travel–which, of course, we have: "FROM node 0, TO node 1."

BE CONSISTENT! The mathematics involved will merely produce a minus sign if you guess the wrong direction for a current or something, but you HAVE to remember what your guess was! And if you start building a branch of a circuit with −0.01, then every element in that branch MUST BE −0.01, except the last one, which is 0.01. You would NEVER mix notation by using a 1.00 (either + or −) in that branch! That would mess up the solution quite thoroughly.

Remember, too, that the active source elements V and I have directions associated with them. Be sure to make those directions consistent with the direction of your branch! If your voltage points from node 0 to node 1 and you choose your branch DIRECTION to be 1.00 (1 to 0), this is fine, but you have to change the sign on the voltage value itself (say, −120 volts, instead of 120), to tell the calculator that this source is oriented oppositely from the direction of the branch. Understand?

The rules are simple: Keep your signs straight and you shouldn't have any problems.

THE COMPLEX VALUE

So much for the location value. Now, as you know, if you roll the stack down once, you will see the REAL part of this element's value (in volts, ohms, or amps, right?). Another [R↓] will show you the IMAGINARY portion (indicated by flag 2); one more [R↓] will show a place–holding value which is not of direct interest, but it's another reminder of the TYPE of element:

4 means an impedance (Z);

5 means a current source (I);

6 means a voltage source (V).

(And, if you need a more direct reminder, you can always turn on alpha mode, and there you'll find the prompt that flashed briefly before stopping with all these values "stacked up.")

And of course, a fourth [R↓] will bring you back to the location value, which is how the stack should be set up when you press [R/S] to continue.

But how do you key in the element values themselves? A word now about complex numbers:

As you recall, you always represent complex numbers with two values. For the purposes of this program, you can choose one of three different forms of these values:

Rectangular (X + jY)   Polar (Z ∠ a)   Physical (R, L, or C)

The program will "speak" any of these three languages, and you can tell it which one you want. To do this, you use another set of keys which are active on the HIDDEN MENU at this point in the program:

[D] will alternately set or clear flag 1 in the display.

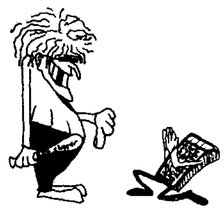
When flag 1 is SET, the program accepts and produces complex numbers in RECTANGULAR form;

[d] will alternately set or clear flag 0 in the display.

When flag 0 is SET, the program "speaks" in POLAR form.

And when NEITHER flag 0 nor 1 is set, the program "speaks" in PHYSICAL form, taking all input as being combinations of R and L or C.

Notice that you cannot have both flags 0 and 1 set at the same time (unless you cheat and do it manually, thus cruelly abusing the confidence of this poor, hardworking machine).



BEWARE of mixing units! The program is maddeningly consistent: If you're inputting impedances in RLC (physical) form, and you come to a V source somewhere in your program, you'd better change the format to something besides "Physical" notation (which is valid only for impedances).

ALSO, if you try to go back and check your inputs later, and you're not in the same format, your values are going to look very weird, and you'll think they're all wrong. Don't be too hasty until you review the values in the correct format (and you can change this format anytime during the editing process)! *YOU HAVE BEEN WARNED.*

Now, give it all a try:

In this first example problem, all the values are likely to be zero, or utter nonsense, so you needn't bother checking any more, really. The pattern is this: For every element, just set up the stack's three important values:

The IMAGINARY part of the element;

The REAL part of the element;

The LOCATION of the element;

and press [R/S]

Then a little menu will appear to let you choose the type of element this is:

"Z I V"

Pick the right key, and you've just drawn this element in your circuit, replacing whatever it was that was being "suggested" as an element. (IF the suggested values ARE what you want– INCLUDING the type as it appears briefly in the display–you just press [R/S], and the program then goes on to the next element without asking you to choose the element type–a quick shortcut).

That's about all there is to know about editing each element in the circuit. The program will go through each of the 7 elements you have reserved, and then send you back up to the Main Menu.

It's taken a lot of words here to explain all this, but after a few elements, you'll pick up the idea. The whole process becomes quite rapid after you're used to it. Go back to page 41 now, and run through those keystrokes, and see if this doesn't begin to make a lot more sense.

ALTERING YOUR CIRCUIT

That's it–you've "built" the circuit. Is it correct?

To find out, choose the [A] key in the Main Menu, which is the Alter routine.

The only difference between the Alter routine and the build routine is that when you Alter a circuit, you cannot change its size; the program doesn't ask you how many nodes and branches–it just goes straight to the frequency and then lets you examine each element, changing the values in the stack as necessary.

And remember! Anytime you're satisfied that the circuit is correct, you can use the [J] ("Jump") key to return to the previous menu (the Main Menu, in this case). You'll be using this (J) key all over the place. It is another member of the HIDDEN MENU, and it's active in every menu.

Actually, in this first example, you should probably review every single element value–to be sure you're doing things correctly and to be sure the program "ain't misbehavin'."

If, anywhere during your reading here, you're not getting the results you think you should, check:

- Your complex notation (i.e. are you and the machine "speaking" the same language?);
- Your method of filling the stack (sometimes it's easy to input the real portion first and the imaginary portion second, since that's the way we are used to thinking about it. But remember! The imaginary portion goes ABOVE the real portion in the stack, so the imaginary portion must go in FIRST.);
- Your program listing–some program line may be mis-keyed.

ADDING TO YOUR CIRCUIT

If you choose the [a] key (press [SHIFT] [a]) from the Main Menu, this is the "add" routine, which allows you to expand your circuit, changing either the number of nodes or elements.

If you INCREASE the number of elements, the editing process will begin with the NEW elements, not at the first element of your circuit.

If you are REDUCING the number of elements, the process will begin at the first element and thereby behave exactly like the build routine–except that most of the values are likely to be worth keeping.

If you don't change either the elements count or the nodes count, you'll be sent back to the Main Menu immediately.

CRUNCHING YOUR CIRCUIT

You may have noticed that any time the Main Menu appears, flag 0 is turned on, indicating that the program is "speaking" in polar notation. You are about to encounter the main reason for this: The results you get when you "crunch" the circuit.

First, you may be interested to know that the crunching process is one in which the program examines each circuit element and decides how that element contributes to the values in two matrices in this equation:

$$Y \times V = I$$

The Y-matrix is a nodal admittance matrix for the circuit, while the I-matrix is the list of all current sources in the circuit. These are the known matrices whose values are accumulated by the program. The unknown matrix is the list of voltages (the V-matrix)–one voltage at each node.

The point of all this is that the crunching process is necessarily time–consuming. You can expect it to take about 20 seconds of loading per circuit element, plus some time for the Advantage matrix routines to actually solve the simultaneous equations (for that's what a matrix equation–like the one above–really is: a series of simultaneous linear algebraic equations, right?) This second part of the solution time varies roughly as the square of the number of nodes. Thus, a 6-node matrix may take about 4 times as long as a 3-node matrix. And you can tell when this part of the calculation is in progress: the goose "freezes" in the display.

So if you tend to get impatient, then ponder this while you're waiting: How long would it take you manually to look at a circuit diagram, figure out what values to load into each of two matrices, then solve the matrix (and since these are complex numbers, a 3-node circuit is a 6 × 6 matrix)??!? Aren't there better ways to spend that time?

So, once you've loaded your sample circuit (the one on page 40), press [C] from the Main Menu, and take the next few minutes to stop and smell some roses....

When the crunching is all done, you'll see the Calculations menu:

"P I VG"

These stand for Power, Current, Voltage, and Gain, respectively. And notice that the Gain key is the shifted version of the voltage key, since both of these calculations are matters of voltage.

V AND G: VOLTAGE AND GAIN

To put it briefly: Choose your nodes first (in From.To format); then select your key:

$$V = V(\text{From}) - V(\text{To}); \qquad G = V(\text{To})/V(\text{From}).$$

In other words:

In the problem we're working on, (page 40) you're asked for the voltages at nodes 1 and 2. Now, as you recall, a voltage is the measure of a potential energy–and that always means it's relative to some reference voltage, which is node 0 (ground): $V_0 = 0$. So the "voltage" at node 1 is really ($V_1 - V_0$); and the "voltage" at node 2 is really ($V_2 - V_0$).

The point is this: To calculate anything from this menu, you need to key in the LOCATION of the nodes you are interested in–using From.To format (same as when you're building your circuit), with the understanding that the answer you get will be a complex number, which, in this case, is:

$$V(\text{From}) - V(\text{To})$$

This means that if you press 1 [C], you'll get $V_1 - V_0$, which is just V_1. (Do this now) But if you press 0.01 (or just .01) and choose [C] (for Voltage), you're asking for the voltage FROM node 0 TO node 1, which is $V_0 - V_1$, or $-(V_1)$.

It's the same idea for Gain [c], except that it's computed as:

$$G = V(\text{To})/V(\text{From}).$$

When a result appears, as usual, the first portion (116.96) will be in the X-register; the second portion (−2.16) will be in the Y-register (and, as flag 0 shows, these portions are in polar notation, which is often the handiest to use when you're working with complex currents and voltages. This explains why the Main Menu sets flag 0 for you). Feel free to use [X◄►Y] and [R↓] as they usually work in this program.

So your answer is:

"The voltage at node 1 is 116.96 $\angle -2.16°$ volts."

Good. But where are YOU?

You are now in the Equations Menu (refer to the Road Map if necessary–on the inside of the back cover). The program automatically sends you down here after each calculation. Here is where you can store results, convert to other complex formats, or do other handy calculations. You'll be reading in detail about these later.

Right now, just notice that you can return to the Calculations menu simply by using Jump ([J]).

When you see: "P I VG"

solve for V_2 by pressing: 2 [C]

"The voltage at node 2 is 111.07 $\angle -137.46°$ volts."

Did you get this result? Now, Jump back once again, ready to solve for the currents you need to know.

P AND I: POWER AND CURRENT

To put it briefly:

Choose your branch first (by indicating the nodes it connects, in From.To format). Then select your key:

$$I = I(\text{From.To})$$

$$P = I(\text{From.To}) \times [V(\text{From}) - V(\text{To})]$$

(Note that this power is the power being dissipated by the impedances in a given branch, NOT the power being supplied by a given source).

The program will search through your circuit elements IN THE ORDER THEY WERE BUILT, until it finds the first occurance of a branch connecting nodes From.To.

If there is more than one such branch, and you want to analyze any of the other parallel branches, just return to the Calculations Menu and ask for the same branch From.To again. The program's search will begin from where it found the previous branch (and, if necessary, even loop around to the beginning of the circuit element list to continue the search).

This means you have to keep in mind the order in which you built the circuit, AND where the program will next begin to search for the branch you specify. This wasn't necessary for node calculations (Voltage and Gain), but it is for Power and Current.

One more caution: ALWAYS analyze a branch in the same direction as that in which it was built. If you built it as 1.02 (From node 1, To node 2), you cannot ask for 2.01 and expect to get a correct answer. The program does NOT recognize 1.02 and 2.01 as being interchangeable names for the same branch! A different name is a different branch!

Even if you know that the current runs counter to the direction of the branch, ask for the current (or power) according to this branch direction. The result will have a minus sign or an angle difference of 180° to indicate the current's opposition to this direction.

So, here are your final solutions and answers. From the Calculations Menu, press:

1 [B]

Result:

"The current in the (only) branch from Node 1 to Node 0 is

0.43 ∠−45.46° amps."

[J] (To Jump back once more to the Calculations Menu)

2 [B]

Result:

"The current in the (FIRST) branch from Node 2 to node 0 is

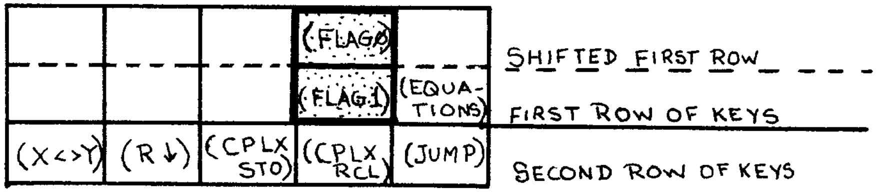
0.37 ∠−141.27° amps."

You're finished with the problem! No muss, no fuss, no dirty dishes....

Of course, now that you're in the Equations Menu, it's a good time to (finally) look at this feature, and also the HIDDEN MENU in its entirety.

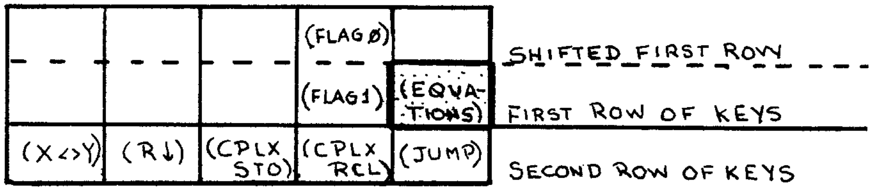
THE HIDDEN MENU

[D] and [d] (Adjust flags 1 and 0)



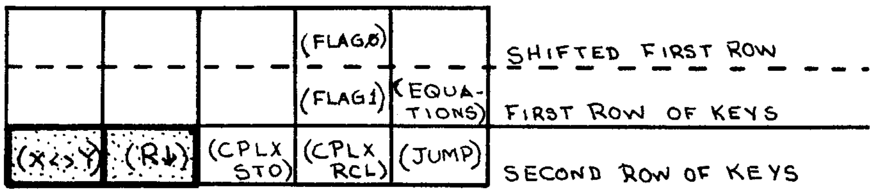
You've already seen these keys. They adjust flags 0 and 1, which control the complex number format. See page 48 if you don't remember how these work. Keep in mind that they are meant to be used only from the Editing or Equations Menus.

[E] (Equations Menu "Side Trip")



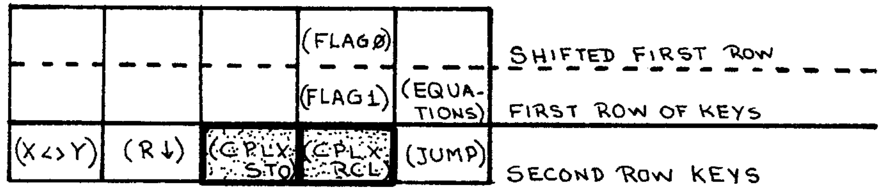
From any menu (except the Equations Menu itself), if you press [E], you will detour to the Equations Menu, as shown on the Roap Map.

[F] and [G] ([R↓] and [X◄►Y])



You've already seen these keys. They do just what they normally do, except that they also alternately set and clear Flag 2, to indicate to you whether you are looking at the first or the second portion of a complex number. Keep in mind that they are meant to be used only from the Editing or Equations Menus.

[H] and [I] ([STO] and RCL] for complex numbers)



Again, these are meant to be used from the Editing or Equations Menus, where you will find them very handy for storing or recalling (to the stack) any complex numbers to be used in building circuits or analyzing your results.

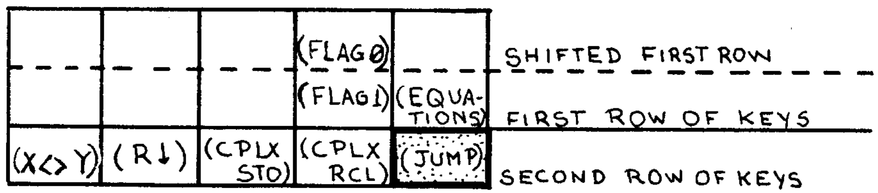
With either [H] or [I], you are using four complex storage registers: A, B, C, and D. When you press [H], a little sub-menu appears, offering you

H: A B C D

Just press the appropriate register's key (A-D), and there you have stored a copy of the complex number that is currently sitting in the X- and Y- registers of the Stack–exactly analagous to the way a normal [STO] works.

[I] works similarly, but it recalls a complex value to the X- and Y- registers, bumping what was previously there up into the Z- and T- registers–just as you would expect.

[J] (Jump to the previous menu)



You've already seen this key several times. It merely sends you from one menu to whatever menu you were working in previously. Of course, using [J] from the Main Menu gets you nowhere at all.

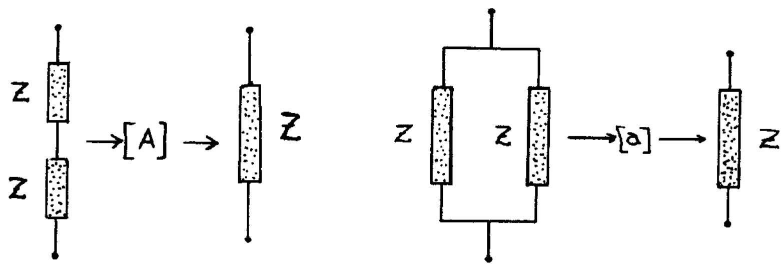
THE EQUATIONS MENU

Looking at the Road Map, you can see that this menu is accessible by way of ANY of the other menus, and naturally, [J] will Jump you back to wherever you were before you took a "side trip" down to this Equations Menu.

After each time you use one of the keys on the Equations Menu, you will be returned back to that menu again. Depending upon what you just did, you may see the result stay in the display or merely flash briefly before the Menu reappears. Either way, you only need to press [R/S] to alternate between the result and the menu.

Here's a brief description of what each of the keys in the equations Menu will do:

[A] and [a] (Combine series or parallel impedances)



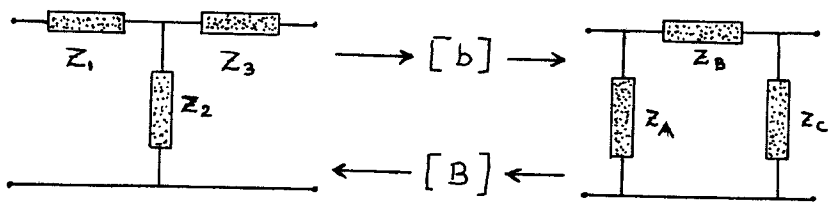
Just as in the editing routine, these keys speak whatever language you are currently working in–Polar, Rectangular or Physical. You load up the stack with four numbers; the top two represent one complex impedance; the bottom two, another. The [A] key will combine these impedances in series (recall that impedances add linearly when in series), while the [a] key will assume they are in parallel (and therefore add them inversely).

BE SURE YOUR TWO IMPEDANCES ARE IN THE SAME COMPLEX FORMAT. The result you get will ALSO be in this format.

[E] and [e] (Complex multiply and divide)

These are very simple to understand: Put in two complex numbers–in the same format–and multiply or divide them in the same manner you normally use for real numbers. Again, the result will be returned to you in the same complex format you used for input.

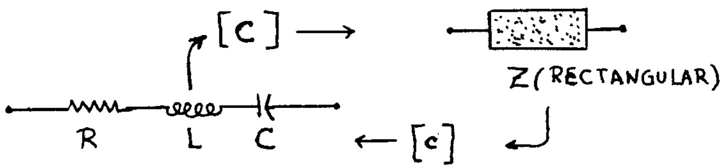
[b] and [B] (Y-to-Delta and Delta-to-Y conversion)



These are common calculations, often necessary to set up simpler circuit configurations. You load three impedances into the Complex Storage Registers A, B, and C, (because three complex numbers won't fit in the stack). Then select [b] to convert a Y configuration to a Delta, or [B] to convert a Delta to a Y.

BE SURE YOUR THREE IMPEDANCES ARE IN THE SAME COMPLEX FORMAT. The resulting three equivalent impedances will ALSO be in this format, and they will be found in those same storage registers A, B, and C (Z_1 goes in A, Z_2 in B, Z_3 in C).

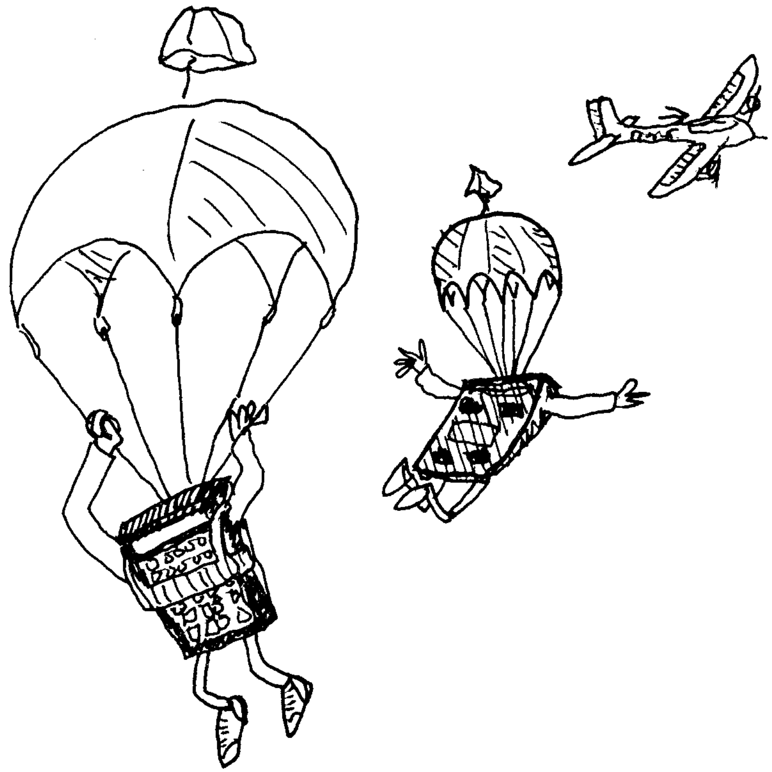
[C] and [c] (Convert C or L values to reactive impedances, and vice versa)



These are the only functions on the Equations Menu that don't try to interpret the format of the complex numbers you load into the stack (you load two complex numbers–four values). It takes you at your word: These are impedances either in Physical (R + L or C) or in Rectangular (R + X) form. It converts the (reactive portions of) the impedances back and forth:

[C] performs C,L–►X

[c] performs X–►C,L



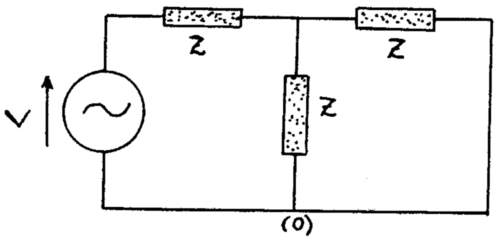
SOLVING TYPICAL PROBLEMS

SOURCE POWER

The power supplied by a source in a circuit is NOT what you are calculating when you press the P key in the Calculations Menu. That key calculates the power being dissipated in the IMPEDANCES of a given branch. The question here is: What amount of power does a given SOURCE supply to a circuit? The answer is:

$$Ps = (Vs \times Is),$$

where Vs and Is are the complex voltage across the source and the complex current through the source (only), respectively. Take an example:



To find the power supplied by the source:

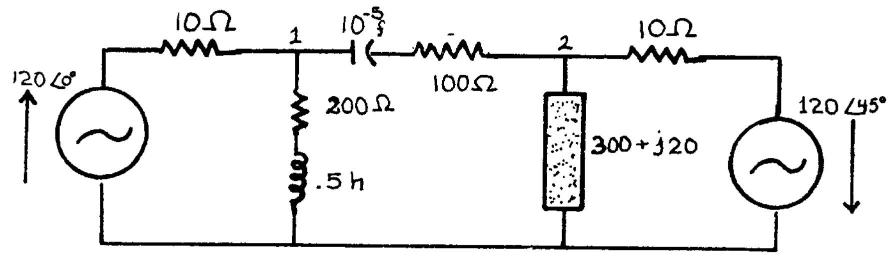
(A) Build the circuit as shown;
(B) Calculate the current through the source's branch;
(C) Multiply your result by the source voltage.

NOTE: The real power is the real portion of the answer, so if you're computing in Polar notation, use the P −►R function to convert your final answer to Rectangular form.

ANOTHER NOTE: If you have an I source, you have to compute the voltage across this source. Since the program will compute voltages only at nodes, it would seem that you would have to build a branch with only this I source (i.e. a node on each end of the source). But the program won't let you do this (it gives 0's for all the answers). Just cheat by including in that branch a negligible resistance. Then you can accurately compute a node which is, for all practical purposes, the voltage across your source.

INPUT (DRIVING POINT) IMPEDANCE

The input, or driving point, impedance at any given pair of points in a circuit is defined as the ratio of a voltage source connected across those two points to the resulting current, WITH ALL OTHER SOURCES IN THE CIRCUIT SET TO ZERO, BUT ALL IMPEDANCES RETAINED.



So if you have a circuit such as the one above, and you want to find the input impedance at nodes 1 and 2, do the following:

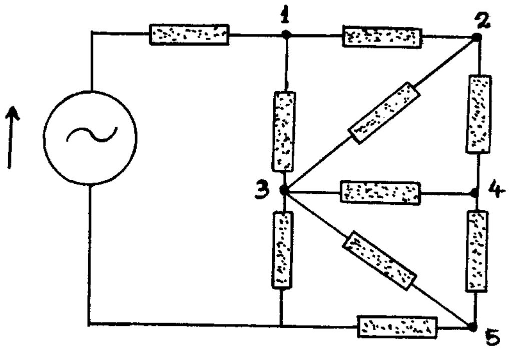
(A) Build the circuit as shown, but either "zero out" or omit the two true sources;

(B) Add a test voltage source across nodes 1 and 2, plus a negligible resistance (1 E-20) in series with this source.
You do this because the program will NOT accept as a branch just a single source without an accompanying impedance (a stipulation which reflects physical reality).

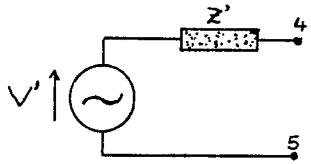
(C) Solve the circuit for the current through this test branch; divide the result into your test voltage to get the input impedance.

THEVENIN'S EQUIVALENTS

Somebody named Thevenin once figured out that it would make life a lot simpler if, instead of having to work with this:



you could work with this:



You can make this substitution PROVIDED that the V' is equal to the open circuit voltage difference across these same two nodes in the original circuit, AND that the Z' is equal to the input impedance at those two nodes.

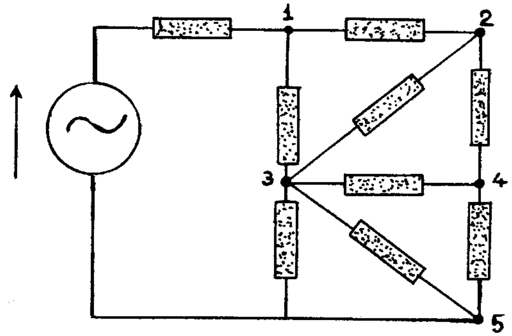
To compute the equivalents for the above circuit:

(A) Build the circuit as is, and solve for $V_{4.05}$. This is the open circuit voltage across the two nodes you're analyzing–the Thevenin's equivalent voltage source;

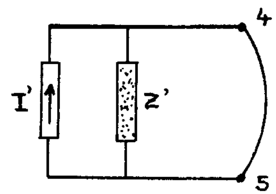
(B) Solve for the input impedance at nodes 4 and 5 (see page 64 if you don't remember how to do this). This is the Thevenin's equivalent impedance, Z'.

NORTON'S EQUIVALENTS

Norton's equivalents are similar to Thevenin's, except that instead of an equivalent VOLTAGE source with a value equal to the OPEN-CIRCUIT voltage between the nodes in question, you're looking to find an equivalent CURRENT source with a value equal to the SHORT-CIRCUIT current between those two nodes in the original circuit. That is, you're looking to replace this:



with this:



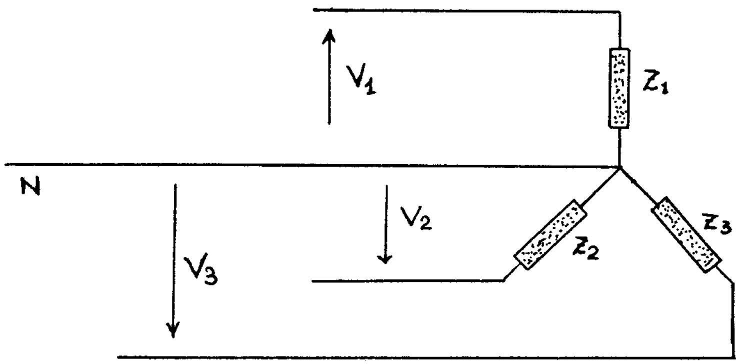
To do that:

(A) Build the circuit as is, except put a negligibly tiny resistor on a branch of its own between nodes 4 and 5. This is the program's equivalent of a short-circuit. Then solve for the current in this branch. This is your Norton's equivalent current source, I'.

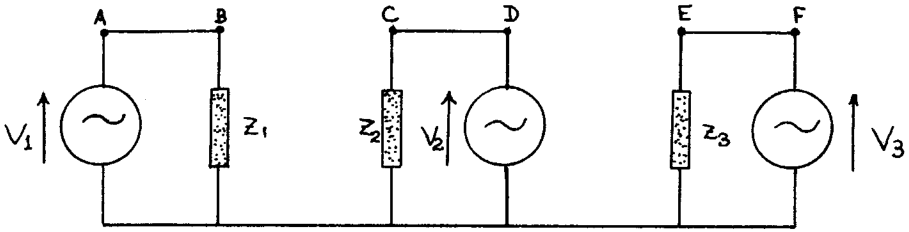
(B) Compute the input impedance at nodes 4 and 5 (see page 64 if you've forgotten how to do this). This is the Norton's equivalent impedance, Z' (yes, it's the same as the Thevenin's equivalent Z'.)

THREE-PHASE POWER

In the course of your studies (or in the study of your courses), you will undoubtedly encounter a lot of problems in three-phase power, which is the common method of wiring for a wide variety of applications. When you use this program to analyze circuits such as the following:



you might want to rebuild the circuit like this, to make it easier to visualize, and to allow you to analyze line currents more easily:

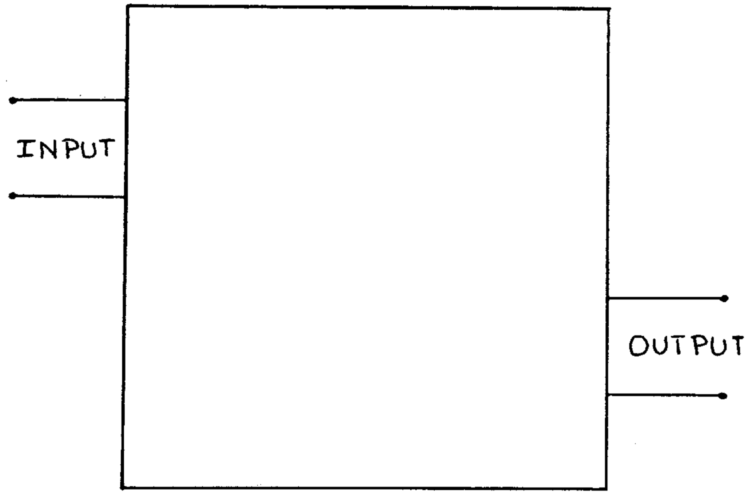


You will have to add negligible resistors between points A and B, C and D, and E and F, then use points B, C, and E as nodes.

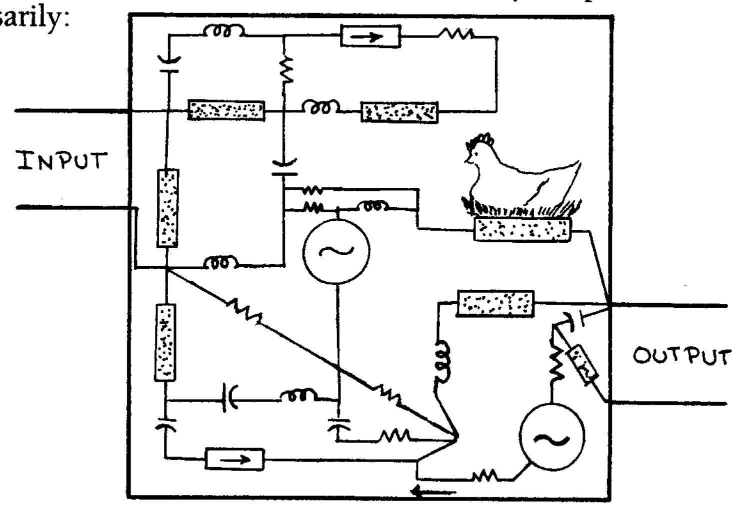
Remember, too, that you can compute Y-to-Delta conversions with the Equations Menu. This comes in handy with such three-phase systems.

TWO-PORT NETWORKS

Two-port networks are circuits that have a characteristic input "port" (a pair of nodes) and an output port:



In between may be a jumble of elements–usually all passive, but not necessarily:



Typically, though, you're only interested in the voltages and currents at the two ports, so use what you know about input impedances and Thevenin's and Norton's impedances to give you whatever quantity you want.

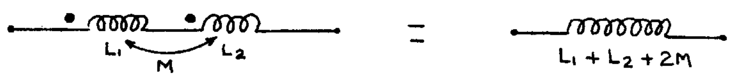
MUTUAL INDUCTION

Mutual induction is a phenomenon where two or more inductors in a circuit have an extra effect on one another. The reasons for this effect are similar to the reasons for an inductor's primary effect on a circuit: Changing currents cause magnetic fields, which in turn, may further change the current (albeit in another direction).

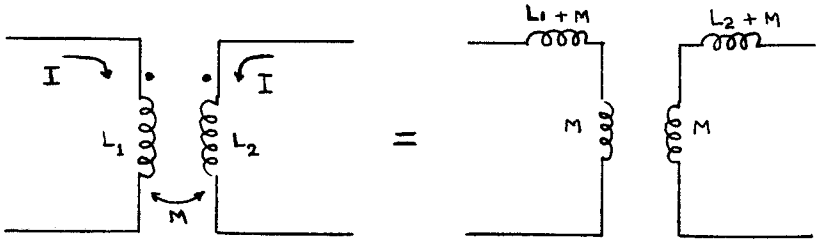
Normally, a coiled inductor works because of its own magnetic field–the field lines of force set up by the changing current–within its coils. With mutual induction, another coil is near enough so that some of its magnetic field ALSO sets up lines of force within the coils of the first inductor. Naturally, the first inductor returns the favor.

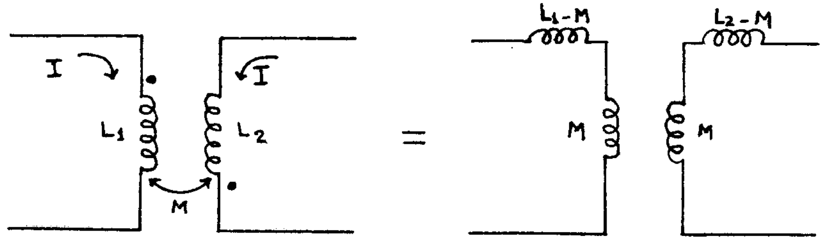
In circuit notation, the common way to denote the directions of these added effects is with a dot at one end of each inductor; there are rules of thumb to determine how these dots signify the effect of the mutual induction, M.

In a series circuit:



In parallel circuits:



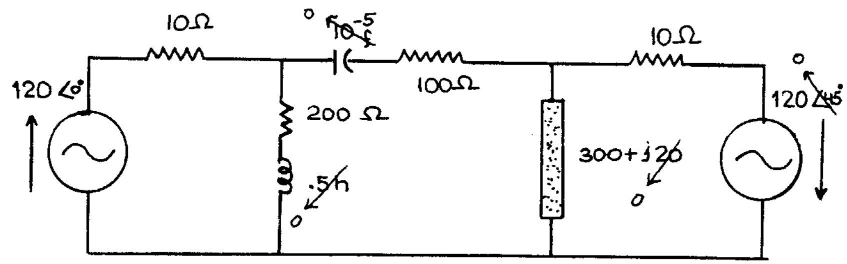


DC CIRCUITS

Direct Current (DC) circuits are easy to build and crunch with this program.

Of course, a DC circuit is simply an AC circuit with a Frequency (F) of zero. But this program won't let you use an F of exactly 0, and in general, you really want to vary F to analyze circuits that do have reactances–not try to model non-reactive (i.e. DC) circuits.

How, then should you crunch DC circuits? Simple: No matter what your frequency is, if all your elements are just plain resistors, then you can use as many IN-PHASE sources (i.e. 0°) as you want. So just build the circuit as you see it, except leave out (i.e. set to zero) any reactive elements and any source phase angles.



The editing routine is tailor-made for you to do this, since each impedance has two parts to it.

That about wraps things up. For quick reference, we have put the Road Map, the Table of Contents, and a troubleshooting page (page 74) in the very back of this book, so if you are lost or you sometimes forget what certain keys will do, try those pages first.

If you are crashing the program, and after reading page 74 you can't find anything you're doing wrong, it may POSSIBLY be an unforeseen feature in the program (also known as a bug). We are, after all, card-carrying members of the species Homo Sapiens, and these things can happen. If you can, try to reproduce the error consistently (so we can diagnose it and let you know the correction(s) to it). Then drop us a line at the address on the title page, describing the entire situation–data and keystrokes–and we'll fix it up, good as new (right after we severely flog the programmer responsible).

But don't let that be your only reason for writing. Even if you never crash the program for any reason, we would like to hear your comments on the program–what you like and dislike about its design, things you would rather we had included or left out, suggestions for other programs–either in circuits or other areas of student engineering, etc. We like to hear from our readers, and we always read our mail. Good Luck in your studies!

If you liked how we wrote the program (and this book), you might also be interested in other applications we have for the Advantage Module (OR in learning how to write HP-41 programs of your own)! Here is a complete list of

OTHER BOOKS FROM THE PRESS AT GRAPEVINE PUBLICATIONS, INC.

→

Please use the nifty order blank on the reverse side of this page. We'll send you (or your friend in need) a catalog describing each of our books in greater detail, and it's free!

OR, if you are so inclined, dial us direct at (503) 754-0583

GRAPEVINE PUBLICATIONS

–*Using your HP-41 Advantage: Electrical Circuits for Students*

–*Using Your HP-41 Advantage: Statics for Students*

–*An Easy Course in Programming the HP-41*

–*An Easy Course in Programming the HP-11C and 15C*

–*An Easy Course in Using the HP-12C*

–Custom Programming Services

"I would very much like you to send me information on all of your books, services, etc."

NAME: (please print)____________________________

ADDRESS:____________________________

CITY, STATE, & ZIP:____________________________

Mail this to: GRAPEVINE PUBLICATIONS, INC.
P.O. Box 118
Corvallis, OR 97339-0118
U.S.A.

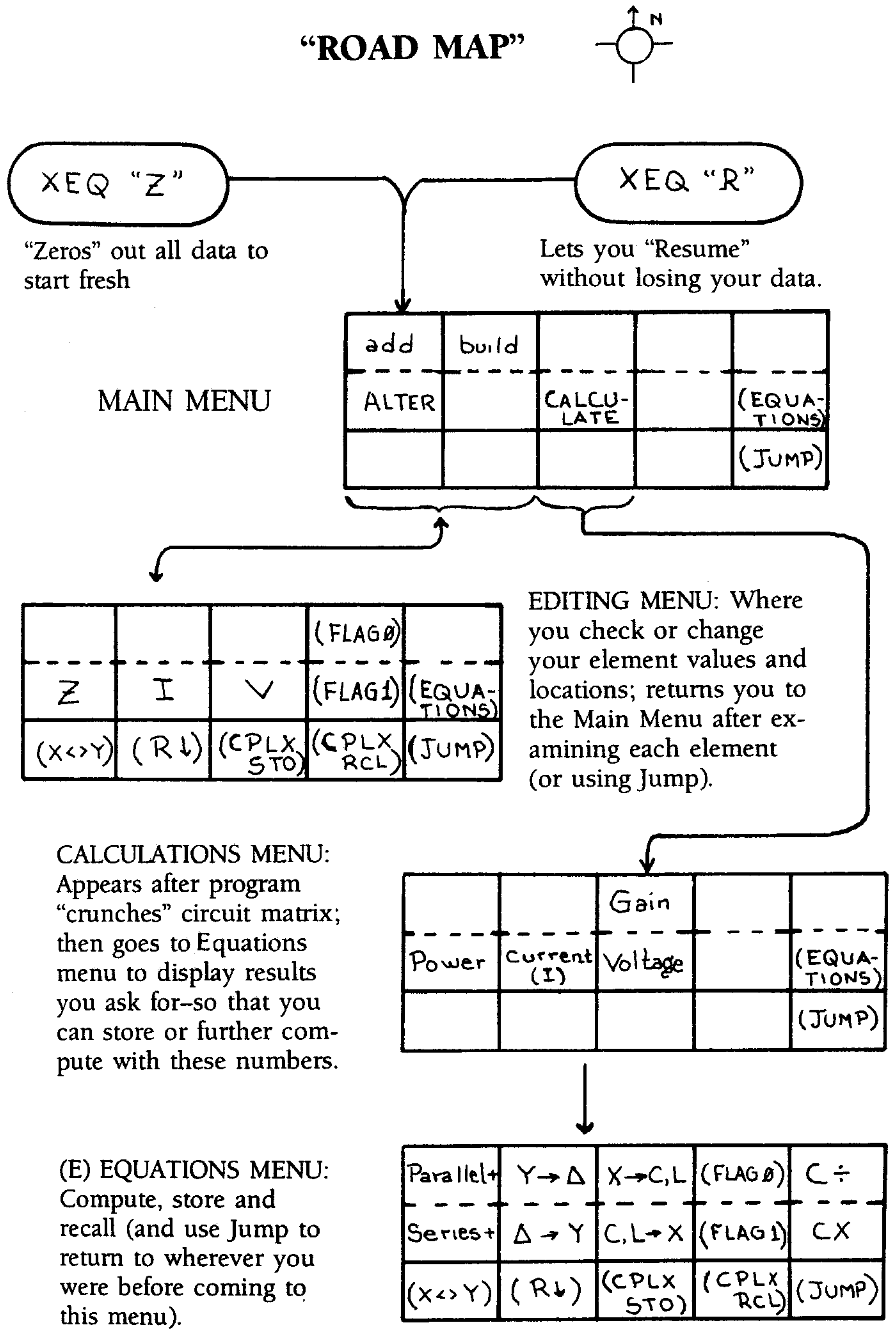
PROGRAM ERRORS AND PROBABLE CAUSES

WHAT'S THE PROGRAM DOING?	WHY IS IT DOING THIS?
"NONEXISTENT"	You don't have your SIZE set high enough for the circuit you're trying to build (see page 43).
"NO ROOM"	You don't have enough room in Extended Memory for the circuit you're trying to build (see page 43).
"DATA ERROR" or "OUT OF RANGE"	You're using a frequency of zero; or you're trying to divide by zero in some calculation; or you have a resonant circuit or sub-circuit.
"END OF ARRAY"	If you're crunching or editing you probably gave the machine some bad data for building a circuit. Check your circuit with the "add" routine (so you can also check the nodes and branches count).
An endless loop – the goose just keeps on flying.	You're in the Calculations menu, and you probably asked to analyze a branch that does not exist: pay attention to "direction" in each branch in the circuit AS YOU BUILD IT. You must ALWAYS refer to a branch in the same direction as that in which it was built.

NEVER panic. You can't hurt the program or the machine. If using [J] after some error message doesn't return you to some familiar point, then use [XEQ] "R" to "Recover" to the Main Menu (without losing your circuit data)!

WHERE TO FIND IT

"ROAD MAP"



"If it's not shown on these menus, don't press the key!"