Notex Command Language
Version 5.2
April 5, 1979

The command language is used in conjunction with command files and can be created and edited with bravo or the edit function in Notex. The language provides the user with most of the facilities avaiable at the console (and many that are note) so that the functions can be prepared in advance, executed without operator error and remembered as long as the user wishes.

The following is a description of the command language. There are some special characters of which the user must take note; space, comma, gets(backward arrow←), upward arrow(↑), semicolon and carriage return. The space is for field separation and the return for command line separation. In general, a space can occur after any command and a return after any command line.

A return is also used to specify data storage whenever input is required. For convenience, a gets(←) character can be used for data input. A comma(,) is used to skip data input fields for the D, W and w commands. The description for the D command contains further explination about the comma. An upward arrow is used to input data but will not advance the address counter. This is a special input to allow many items to be inputted into a given port.

A semicolon, when not in a string, signifies to Notex to skip the rest of the information in the line until after the next carriage return. This is obvously to allow the user to imbed comments in the command file.

Notex allows some primative forms of arithmetic and logical operations when hexidecimal values are being inputted. The operations are executed in the order in which they occur(the same fashion as your TI programmer calculator). i.e. 1234+555/B0*100 will render 2200 as an answer.

There is one special operator($) which is used by some of the commands to specify one of 16 internal registers which only the command files can use. The arithmetic and logic operators are:

plus	+
minus	-
multiply	*
divide	/
and	&
or	%
exclusive or	@
not	~
register	$(This operator is explained in command P).

The language is not very forgiving. If any other characters appear in the command field other than the ones specified below, the command line execution will automatically abort and the Start indicator will be turned off.

Command line	**Comments**
A counter←from←to←	counter=0 thru 7
A4 counter←loc←←mask←	counter=0 thru 7
A6 counter←port←←mask←	counter=0 thru 7
A counter←value←	counter=8 thru F
A4 counter←loc←←	counter=8 thru F
A6 counter←port←←	counter=8 thru F

Assign counter c the starting value From and the ending value To. There are 8 register pairs (c=0 thru 7) which can be assigned as counter/limit or word/mask registers working in conjunction with the J(jump) command. Once a register pair has been assigned a starting and ending value(or word and Mask value), its control file position is remembered. Subsequent jump commands which count the starting value up or down, or which compare the word and mask values will jump back to the assign position until the starting value reaches the ending value or the word/mask values dont meet the jump conditions. The count direction or jump conditions are specified by the delta parameter of the jump command**(Refer J).**

It should be noted that to use the register pairs as counters, a space must follow the A(assign) command. To use the registers as word and mask conditional pairs, the assign command must be followed by a 4 to load the register from a location in the 8086 or a 6 to load the register with the contents of a port.

There are 8 other registers (c=8 thru E) which can also be assigned with the assign command, but they are not conditional and do not work with the jump commands. Register F is a special register which can be set via the assign command but is also set to the starting address of the most recently loaded MB file. This action is automatic and overrides any previous setting of the register.

B ←	Boot with Diagnostics
B2←	Boot without diagnostics

Performs an initial Boot with or without diagnostics.

C A←

Call is made to address A. **This command must be followed by a puase (P) command.**

D A←n←v1←v2←...vn←	Word mode
D2 A←n←v1←v2←...vn←	Byte mode

Deposit n values (v1 thru vn) into address A thru address A+n-1. If the user wishes to skip any value v1 thru vn, a comma(,) can be used instead of a gets(←).

E A←n←	Word mode
E2 A←n←	Byte mode

Examine contents of address A thru A+n-1.

e ←	Open notex.er at beginning
e2←	Open notex.er at end

Compliment state of the Error File flag. If the flag is turned on, The error file 'Notex.er' will be opened at the begining or the end. If the flag is turned off, the file will be closed. If notex.er is opened at the begining, all previous error data is to be lost. If it's opend at the end(2), all the previous data is preserved.

F E←A←n←	Find match for Word E
F2 E←A←n←	Find match for Byte E

Find all the matches for element E from address A thru A+n-1.

H

Compliment state of the Halt on Errors flag.

h

Compliment state of the Halt between Errors flag.

i ←	Load File
i2←	Verify File
i6←	Load & Verify File

Load and/or verify the selected file. **This command must be preceeded by a load (L) command.**

J counter←delta←	if delta=>0 jump if from<to
	if delta<0 jump if from>to
J4 counter←delta←	if delta#0 jump if word&mask#0
	if delta=0 jump if word xor mask#0

Count the specified counter up and jump so long as from less than to(delta greater than or equal to 0) or down and jump so long as from is greater than to(delta less than zero); or compare counter with mask and jump if word and mask not equal zero(delta not equal zero) or jump if word exlusively or'ed with mask not equal to zero(delta equal to zero). **This command must be preceeded by a Assign (A) command with the same object counter. The counters must be from 0 thru 7. They can not be greater than 7.**

L 'Filename.mb'←i*←

Select file 'Filename.mb' for loading and/or verification. Note: the file 'Filename.mb' must have an .mb extension. **The insert (i) command must follow to complete the load. The asterisk following the i indicates a space, 2 or 6.**

l

Compliment the state of the Loop flag.

M A←n←B←

Move the contents of address A thru A+n-1 to B thru B+n-1 on a byte by byte basis.

P

Pause until the 8086 becomes ready to recieve another command or the user strikes the escape key(this terminates the control file execution). **This command must be preceeded by a Call (C) command.**

p S←	Print string S
p1 S←	Print return then print string S
p2 S←	Clear the user display and print string
p4 $c←	Print contents of c(c=0 thru f)
p5 $c←	print return and then contents of c

Print string S or counter c with specified formating in data display area.

Q

Quit. Requires a return for confirmation.

R A←n↑	Normal mode
R1 A←n↑	Single step mode

Read port A, n times. Single step mode is provided to allow the user time to view the subsiquent values being read from the port by requring the user to prompt(strike the space bar) between each value.

r

Run or proceed to the location specified in 8086 register IP.

S 'Filename.cf'←i←

Start command file 'Filename.cf'. Note: the file 'Filename.cf' must have a .cf extension. **There can be at most three command files opened at any time. i.e. file1.cf can start file2.cf which can start file3.cf and file3.cf must terminate if file2.cf is to start another command file.**

s E←A←n←	Spread Word

s1 E←A←n←	Spread incrementing Word(delta=2)
s2 E←A←n←	Spread Byte
s3 E←A←n←	Spread incrementing Byte(delta=1)
s4 E←A←n←	Spread alternating Word
s5 E←A←n←	Spread random Word

Spread the various types of patterns from location A thru A+n-1. The following explains the action in more detail:

space
E is a word(16 bits).
1
E is first word. All succeeding words incremented by 2.
2
E is a byte(8 bits).
3
E is first byte. All succeeding bytes incremented by 1.
4
E is first word. All succeeding words are the one's compliment of each other.
5
E is first word. All succeeding words are random.

W A←n←v1←v2←...vn←
Write n values (v1 thru vn) into ports A thru A+n-1.

W A←n←v1↑v2↑....vn←	Normal mode
W1 A←n←v1↑v2↑....vn←	Single step mode

Write n values (v1 thru vn) into port A. Single step mode is provided to allow the user time to view the subsiquent values being written to the port by requiring the user to prompt(strike the space bar) between each value.

w R←n←v1←v2...vn←
Write n values (v1 thru vn) into Registers R thru R+n-1. R can be 0 thru D.