

**Notex Program Description**  
**Version 5.2**  
**April 5, 1979**

This program serves as a tool for aiding in the initial system integration and subsequent acceptance testing for the NoteTaker system under the control of an Alto. The program operates via the Hytype interface with an appropriate debugger module plugged into the NoteTaker.

Notex allows the user to load programs into the Note Taker memory, to access the programs and to display messages from the programs. Notex also provides a wide range of utility debugger features for the user which are described in other sections of this document.

The Notex user interface is patterned after a maintenance panel which has some alpha numeric capability. The panel buttons are manipulated by use of the mouse, all selections are made with the mouse, and numerical input is done via the keyboard.

Figure 1 is a description of the user interface display. The display is partitioned into four sections. The top section is called the Status Display Section and contains operational status from NoteTaker. The second section is called the Command Select Section and contains the operational command selections which are selected with the mouse. The third section is called the Debugger Section and contains related debugger selections and parameter input all of which are selected with the mouse. The fourth section is called the Data Display Section and this is where all the messages to the user will appear.

Status Display Section

This section indicates the current state of the Note Taker machine if the Alto can communicate with the NoteTaker debugger.

There are six items displayed and they are:

**Control Program**

Displays the name of debugger(iop or ep).

**Diagnostic Program**

The name of the most recently loaded Ram Diagnostic Program.

**Program Counter**

The next location to be accessed by Note Taker. This is only true when the Note Taker is in a halted state. Notex does not display the counter dynamically.

**Machine State**

The Run or Halt state of Note Taker.

**Flag Register**

The contents of the flag register by bit position when the Note Taker is in a halted state.

Command Function Select Section

This section contains the basic command functions for the Note Taker system. The first row of command functions are to control the test environment and the second row provide basic controls of the test program and its data. Since these controls are so germane to the operation of the test programs, it might help to read the first few paragraphs of the Ram Diagnostic Program description prior reading this portion of the document.

Command Function selection is done by pointing the mouse to a command box and then click a mouse button. Some of the commands have additional options depending on which mouse button is clicked. The explanations below describe the commands and their options.

There are seven environment command functions and they are:

#### **Quit**

This command terminates the session, dumps the Error File to the local printer(if there is one) and requires user confirmation.

#### **Loop**

This command allows the user to loop on the current test module.

#### **Halt on Errors**

This command allows the user to halt the ram diagnostic if an error is detected.

#### **Halt bewteen Tests**

This command allows the user to halt between test modules.

#### **Clear**

This command allows the user to clear the Data Display area and clean up any inconsistencies in the notex display. There are two additional options with this command:

middle	Reset all Break Points.
right	Zero out all registers.

#### **Error File**

This command allows the user to open or close Notex.er and requires confirmation for opening the file. If the file has been opened during a session, it will be automatically printed out on the local printer after the user Quits out of Notex. There are two options with the command:

not right	Open Notex.er. All previous data is lost.
right	Append Notex.er. Preserves all previous data.

#### **Registers**

This command allows the user retrieve the register contents from the 8086 and put them into the displayed registers or visa versa. There are two options with the command:

left	Retrieve 8086 register contents.
not left	Set 8086 register contents with the displayed values.

There are ten Program control functions and they are:

#### **Run**

This command transfers control to the 8086 processor directing it to start accessing 8086 code at the location specified by the CS and IP register contents. The run/halt state of the 8086 is indicated by the Machine state box in the status display section. If the machine is already in a run state, the operation will be ignored. The contents of the 8086 internal registers are displayed in the debugger display section and the .status display sections.

#### **CallSub**

This command transfers control to the 8086 processor directing it to start accessing 8086 code at a location specified by the user. The code to be accessed must terminate with a return(RET) instruction in order for Notex to regain control. Notex will ask the user to input the starting address. The contents of the 8086 internal registers are not displayed in the debuuger display section.

#### **Step**

This command transfers control to the 8086 processor at the location specified by registers CS and IP and will execute only one instruction and then return control to Notex. The contents of

the 8086 internal registers are displayed in the debugger display section and the status display sections.

#### **Nmi**

This command regains control from the 8086 processor if the machine not already in halt mode. The run state can be initiated by a run or callsub command. The contents of the 8086 internal registers are displayed in the debugger display section and the status display sections.

#### **Boot**

This command sends a reset and a boot signal to the 8086 and the prints out a success or fail message to the user. After a boot, the machine should be in a ready state indicated by the 8086 ready message above the diagnostic selections box. A successful boot initializes the registers contents to zero and esablishes communications with the 8086 debugger. At this point the user can begin to use the interface. **If boot fails, the interface can not be used.**

#### **Load**

This command causes a list of the available dot MB files to appear in the Diagnostic Selections window so the user can select one to be loaded into the 8086 ram. The Diagnostics appear twelve at a time and bugging the scroll bar scrolls the list of programs. Selection is done by bugging the desired program name. If the desired program cant be found or if the user decides not to make a selection, bugging the load again will restore the interface to normal.

#### **Examine**

This command allows the user to display the contents of the 8086 rom and ram in the data display area. The command responds by asking the user to input a starting address and a length(in bytes). If the length is greater then the starting address, it will be interpreted as an ending address. The command has two options:

not right	The display will be in word mode.
right	The display will be in byte mode.

#### **Deposit**

This command allows the user to alter the contents of the 8086 ram and displays the results in the data display area. The command responds by asking the user to input a starting address and a length(in bytes) followed by the data. The data is delimited with a carriage return. If the length is greater then the starting address, it will be interpreted as an ending address. The command has two options:

not right	The deposit will be in word mode.
right	The deposit will be in byte mode.

#### **Read Port**

This command allows the user to display the contents of any of the 8086 ports in the data display area. The command responds by asking the user to input a starting port address and a length(in bytes). If the length is greater then the starting port address, it will be interpreted as an ending address. If the length is delimited with an upward arrow(↑), the same port will be read length/2 times. there are two options with this command:

not middle	The display will be as described above.
middle	A prompt is required between each displayed word.

#### **Write Port**

This command allows the user to alter the contents of any of the 8086 ports and displays the results in the data display area. The command responds by asking the user to input a starting port address and a length(in bytes) followed by the data. The data is delimited with a carriage return or an upward arrow(↑). The upward arrow prevents the port address from incrementing

thus allowing the user to deposit several different values into the same port. If the length is greater than the starting address, it will be interpreted as an ending address.

#### Nmi

This command transfers control to the 8086 processor at the location specified by registers CS and IP and will execute only one instruction and then return control to Notex. The contents of the 8086 internal registers are displayed in the debugger display section and the status display sections.



# NOTEX INTERFACE BOARD INPUT/OUTPUT

