SPERRY UNIVAC

INTERCOMMUNICATION

TO: Sales Managers
S. A. Managers

FROM (NAME & EXT): J. L. Benson, 560

LOCATION & DATE: Washington , D. C. 9/3/76

DEPARTMENT & M.S.: Marketing

CC:

SUBJECT: UTS 400 Programmability

Attached for your information is an overview of UTS 400 Programmability. This is a general description of the capability, and it is presently being reviewed by the different groups within Univac. Since only a limited amount of data is available on the subject, I felt you should have the review copy for your use. The information within the document is accurate and can be used.

After all the parties review the document, it will be published as a Product Bulletin.

J. L. Benson

JLB:emf
Enclosure

UTS 400 PROGRAMMABILITY (OVERVIEW)

Programmability is a primary attribute of the UTS 400. It is this capability that sets it apart from its predecessors, the U 100 and U 200. Through programmability, the basic UTS 400 functions contained in the firmware can be extended, enhanced and/or modified to increase its ease of use. UTS 400 applications can be designed to function in a unique production environment.

User programming is required only when unique functions are desired. Then only the functions required to be added or changed must be coded by the user. Also, it is important to note that the Univac approach to programmability allows user use of the UTS 400 functionas as provided by the firmware.

As noted in figure 1.1. Univac programmability approach as compared to other offerings of intelligent terminals does not use user storage for a protocol and basic operations of the UTS 400.

This approach is unique among intelligent terminals. In some, a programmable capability is used to provide the overall functionality of the terminal. The basic functionality of the UTS 400 resides in firmware; the purpose of user programmability is to increase or modify the terminal capabilities.

The firmware functions, housed in read-only memory (ROM), are always available in the UTS 400. Firmware does not need to be loaded before the terminal system will function, nor is it destroyed in the event of power failure. Further, it does not compete for random-access memory (RAM) locations which could be occupied by the user program. Thus, the full amount of storage, up to 24,000 bytes, can be devoted to the user.

Enhanced Firmware Functions

User programmability, then, complements the firmware by building on its functional base. The firmware provides the communications protocol, enables dialogue with the host processor, provides for transactions between the UTS 400 and its peripherals, and supplies an operation corresponding to each keyboard input.

User programs cannot change the communications protocol; they can use the firmware-controlled protocol as a conduit to send data to and receive data from the host. A similar relationship exists with respect to the peripherals. Since the user programs need not be concerned with such details of terminal operation, programming becomes simpler and easier.

## Host/UTS 400/Peripheral Relationships

Although the user programs are executed by the UTS 400, they are generated on a host processor and conveyed to the terminal by way of a communications line. (This procedure is called downline loading.) User programs cannot be generated at the UTS 400. The purpose of the UTS 400 is to generate, format, and manipulate data in a communications environment. The user programmability of the UTS 400 should not be construed to mean that it can be used to transform the UTS 400 into a freestanding processor system for small business applications. Rather, it should be considered in the light of the definition of distributed processing wherein a greater part of an application is performed in the terminal subsystem relieving the host of these duties and achieving substantial reductions in communication s overhead and line cost.

Since programs are not generated at the terminal, terminal peripherals are not required to generate, load, or store user programs. However, if the application warrants storage peripherals, or if local retrieval is desired, the UTS 400 can be so configured. In this case, the host-generated programs can be downline loaded and directed to the peripherals, from which they can be retrieved and executed as needed.

## Software Package

Programs for the programmable UTS 400 may be generated by means of a SPERRY UNIVAC software package, which is supported on SPERRY UNIVAC data processing systems. Programs created by the package can be arrayed in the host processor and called to the terminal system in any one of three ways: by a terminal operator, by a program running in the UTS 400, or by an application program running in the host. In any case, the host will take the request, obtain the desired program, and downline load it to the terminal system.

## Examples of User Program Tasks

The following list gives you some idea of the types of user programs that could be executed by the UTS 400:

- Data validation (such as restricting entries, range checking, comparison checking).
- Data formatting/reformatting (such as changing the order of information, or excluding information).
- Data creation

- Data editing
- Arithmetic operations
- Security checks (such as the use of passwords or of code sequences).
- Highlighting invalid entries (or items requiring operator attention).
- Prompter sequences (for operator guidance).
- Text compression on transfer

For example the UTS 400 can be programmed to intercept keyboard-entered characters and cause a specific key (or keys) to initiate a special editing function, such as clearing a designated column of data on the screen, clearing selected fields while leaving others unchanged, loading constraints into selected fields, or duplicating the contents of a particular field into one or several different areas of the screen. (Execution time for user programs will depend on the complexity of the operations and the amount of data involved in the operations. In general, the time to perform operations of the types just mentioned should be of the same order of magnitude as the time required to do firmware-supplied edit functions, such as inserting or deleting characters in the display.) Programmability gives a new dimension to Univac terminal applications. As the term implies, it gives the user the capability of modifying and enhancing the capabilities of the UTS 400 to fit his specific needs, increasing the effectiveness of the total computer system.

## Firmware

The function of firmware in the UTS 400 is to provide the instructions (Code) in ROM which when executed give the UTS 400 its functional characteristics. When an action is initiated by the keyboard operator, a host or a peripheral, a sequence of instructions is executed. These actions are seen by the firmware as a hardware interrupt which temporarily stops the current operation to give attention to the present action. All functions performed by the firmware are started by an interrupt. The firmware is written to interpret the source of the interrupt, get the data from the hardware interface associated with the interrupt, determine necessary action based on the data, and schedule the execution of that action.

When an interrupt occurs, the UTS 400 disables further interrupts. This action gives the firmware a chance to handle the first interrupt without interference from another. Since interrupts can be lost if they are disables for a long period of time, the firmware is programmed to do as little handling as possible while interrupts are disabled. In some cases, all the handling is done while interrupts are disabled; however, the processing is usually passed off to a piece of code which will be executed with interrupts enabled.

The firmware is constantly cycling awaiting interrupts which initiate some activity. The activity performed is determined by the type of interrupt and the data associated with it.

In this description, the code executed because of an interrupt up to the point where interrupts are reenabled is termed interrupt code. The code which finishes the processing with interrupts enabled is termed noninterrupt code. Both types of code are associated with each hardware interface and the total of these types for a given interface is termed a handler. When an interrupt occurs, the address of the next instruction to be executed is saved. This address is known as the return address and is important because the UTS 400 must eventually get back to that point in the code to allow the interrupted activity to be completed. When the interrupt code is ready to return to the noninterrupt code, it executes an instruction which causes the CPU to re-instate the return address and begin executing instructions at that address. As noninterrupt code is executing, it could at any point be interrupted, interrupt code could be executed, and the noninterrupt code returned to.

The UTS 400 firmware contains a scheduler. The scheduler maintains the serial nature of UTS 400 functions and ensures that interrupt code does not affect functioing of the noninterrupt code. The scheduler is responsible for linking the processing of an interrupt from the interrupt code to the noninterrupt code. It also serves as the idle loop of the system since the CPU is always executing instructions whether or not there is a function to perform.

The power on confidence (POC) is a piece of firmware executed only on initialization of the UTS 400. The POC performs tests on the UTS 400 hardware and initializes the UTS 400 to a known state.

The UTS 400 firmware also contains parity error handler. When a hardware parity error is detected, an interrupt occurs and the parity error handler is executed. Its function is to determine which hardware modules detected the error and to increment error counts. It also determines whether the error can be recovered from and executes recovery procedures if it is of that type.

The UTS 400 firmware then, is made up of the power on confidence test, the parity error handler, the scheduler, and the interface handlers. The interface handlers are known by the interface they are associated with: the keyboard display handler, the communications handler, and the peripheral handler. These interface handlers provide keyboard display operations, the communications protocol, and the peripheral operations.

## Interfaces

The UTS 400 offers several interfaces for a user program. The main interfaces are:

1. Keyboard
2. Display
3. Communications
4. Peripherals

Each interface consists of:

1. Entry points to the user program
2. Entry points from the user program to the firmware
3. Flags, pointers, etc.

### Keyboard

There are multiple entry points into the keyboard handler depending on the type of keyboard input present. Separate entry points are provided for the normal alphanumeric keys (A, B, etc.), the function keys (insert, delete, etc.), and the program attention keys (F5 through F22). The scan keys (up, down, left, right) and the release of the scan keys also have separate entry points.

There are entry points back to the firmware for each type of keyboard input. The firmware, upon receipt of a keyboard input such as the letter A, gives the user program control at the entry point corresponding to the type of key activated (normal alphanumeric, in this case). The user program can then:

1. Pass the character back to the firmware for normal processing (vie the firmware control at the normal keyboard input entry point).

2. Ignore the character (give the firmware control by going to the dispatcher).

3. Use that character to initiate a unique operation as defined by the user program and go to the dispatcher on completion.

4. Perform processing and then go to a different point for firmware processing.

Note that if no user program is loaded, normal firmware processing will automatically occur.

The display appears to the user program as a block of RAM memory. This block is fixed size regardless of the screen format of the display (24 lines of 80 characters, 12 lines of 40 characters, etc.). Different formats are accomplished by the firmware including or excluding certain parts of the memory. In so doing the firmware uses parameters such as the memory address of the start of the screen and the end of the screen as well as an identification of the screen format.

The user program can address the display memory as well as to the parameters mentioned above. In addition, the user program can address the other pointers and flags which contain items such as the cursor address, the control of the indicators and audible alarm, the beginning of the next line, and so on.

The user program then, can place information on the CRT screen, read it, control the indicators and audible alarm, move the cursor, establish fields via the field control character, etc.

Communications

The firmware provides the user program with the ability to send text to the host and gives the user program control upon receipt of text from the host.

Text is sent from the UTS 400 by preparing the information to be sent in the screen memory or a portion thereof, and passing a transmit to the firmware at the appropriate return point.

(1) When text is received from the host the user program is given control at the specified entry point. The user program can then operate as desired on that screen of information.

In terms of pointers and flags, the address of this particular UTS 400 (RID and SID), the DID or the host operation, the type of transmit being used, etc., are all available to the user program.

Peripherals

The firmware provides the user program with an interface to the peripherals. Through it the user program can send data to the peripherals, retrieve from the peripherals, perform searches, etc.

In order to perform an operation, the user must place information related to the operation (peripheral desired, mode

(1) Program RAM or the screen by-pass memory may also be used for receiving or sending data by the user.

memory then pass the initiation [illegible]

of data flow, search address if not on screen) in locations and pass the initiation screen by pass or RAM command (print or transfer) to the firmware. Depending on the operation, data is taken from the screen and sent to the peripheral, brought from the peripheral to the screen, or the control operation such as searching, is performed. Upon completion the user program is given control at an entry point for peripherals and a completion status related to the operation is placed in a location so that the user program is aware of the success or failure of the operation.

Program Generation

So as to not require a diskette and/or excessive amounts of random access memory the UTS 400 offers no assembly or compilation capability at the terminal. As such programs must be assembled or complied at the host system and then downline loaded.

Programs for the UTS 400 can be generated using either an assembler or a compiler. The assembler, MAC-80, allows programs to be written using a symbolic representation of the UTS 400 microprocessor instructions. This program is then assembled in the host. The compiler, PL/M, allows user programs to be expressed in higher level statements. This program is then compiled in the host. In either case the program generation process is essentially identical to that for generation of programs for use by the host. That is, the program is written in the form of program steps or instructions as dictated by the host package (compiler or assembler) being used. These are fed to the host using whatever file entry media it supports (cards, online CRT, etc.). Once into the host this file, which is the source program, is either assembled or compiled to produce another program file which can be given to a loader for sending downline.

Since a user may desire that more than one program be available, the compiled or assembled programs are placed in a library on the host, from which they can be downline loaded.

Downline loading a program can be initiated:

1. By the operator
2. By a program running in the UTS 400
3. By a program running in the host

Regardless of origin the request for load contains an identification of the desired program. The host obtains the program from the library and passes it to the loader which in turn downline loads the UTS 400. Special code sequences within the information sent allow the terminal to distinguish a program from data and as such, allow the program to be placed in program memory for execution.

Programs can be down line loaded into UTS 400 memory or, optionally placed on UTS 400 diskette. Once on diskette, programs can be loaded into the UTS 400 for execution.

In terms of programming aids, utility routines are provided to facilitate program generation and up line recall of the program is provided for debugging purposes. Local memory dumps, register dumps, memory and register modifications, and execute with selected dumps are all part of program development aids.

The utilities which are written in MAC 80, can be included in the source program to simplify program generation by freeing the user programmer from details of operation he would otherwise have to consider. A variety of utilities are provided. A preliminary list of library utilities to be provided are:

Utility Library
- Function Names and Groupings
- Utility Library Function Calls
- Problem Programs written in MAC 80
- Problem Programs written in PL/M
- Parameter-List Details
- Return Condition Code Bytes
- Packed Decimal Number Representation

Arithmetic Routine Details
- Add
- Add Packed (ADDP)
- Zero and Add Packed (ZAP)
- Add Binary (ADDB)
- Subtract
- Subtract Packed (SUBP)
- Subtract Binary (SUBB)
- Multiply
- Multiply Packed (MUL)
- Multiply Binary (MULB)
- Divide
- Divide Packed (DIV)
- Divide Binary (DIVB)
- Compare
- Compare Logical Characters (CLC)
- Compare Packed Decimal (CPD)

Data Manipulation Routine Details
- Move
- Move left-right (MOVE)
- Move right-left (MOVRL)
- Edit and format for output (EDIT)
- Edit Pattern Characters
- The Field Resulting from the EDIT Operation
- Edit Pattern Length
- Edit Sign Considerations
- Justify
- Left Justify (LJUS)
- Right Justify (RJUS)

Conversion Routine Details
Translate Characters (TRANS)
Packed Decimal
Pack Decimal (PACK)
Unpack Decimal (UNPK)
Pack Characters w/Trailing Sign (PACKT)
Unpack to Characters w/Trailing Sign (UNPKT)
ASCII
Convert Binary -ASCII (CVBTA)
Convert ASCII-Binary (CVATB)
Convert Hex-ASCII (CVHTA)
Convert ASCII-Hex (CVATH)
Multiple Bases
Convert Any Base Number to Binary (CVNTB)

Debugging Aids Routine Details
Debug Monitor (DEBUG)
Debug Monitor Execution
Debug Monitor Operation and Commands
B Command (BNPF OUTPUT)
D Command (DISPLAY DATA)
F Command (FILL MEMORY WITH CONSTANT)
G Command (GO TO)
H Command (HEXADECIMAL ARITHMETIC)
M Command (MOVE MEMORY)
S Command (SUBSTITUTE MEMORY)
X Command (EXAMINE AND MODIFY REGISTERS)

Screen Management Routine Details
Field Generate
Write to Field
Read from Field
Clear Screen
Clear FCC's

File Management Routine Details
Write to Printer
Read from Tape
Write to Tape
Read from Diskette
Write to Diskette

## Application Brief

In order to explain the application of a UTS 400 in an operational mode, the following is a description of a typical application which involves the UTS 400, its peripheral subsystems as well as transactions with the host.

It is assumed that an order entry application has been developed for an automobile parts distributor. The customers of this distributor are retail and wholesale outlets for auto parts. Customers will call the distribution center order office using a toll free number and will be connected with one of a number of order entry clerks who are using a UTS 400 cluster for this application. The person placing the order will provide the operator with either the customer name or customer number. Upon keying this data into the UTS 400, an interaction with the host will occur which verifies the customer name/number, ascertains credit rating and if verified will provide to the UTS 400 subsystem the full customer name, address, credit limits, discount ratios, and any special shipping instructions, etc.

After the customer name has been verified the customer will provide the stock number and the quantity to be ordered. At the completion of each entry a program attention key will be depressed. This will cause an interaction with the host. The host will receive the selected stock number - verify it, recognize any supersessions and supply to the UTS 400 information related to the item such as; description, cost, weight. Note that in this application the operator can be keying additional items while the verification and communication with the host are occurring. As each item is verified the information supplied by the host will be displayed on the screen, and be written on the diskette subsystem.

When the host has supplied all of the requested data the operator will verify the item description, any error, or out of stock conditions with the customer.

When all items of an order have been confirmed a program attention key will be depressed, this will cause the local UTS 400 user to multiply each item unit cost times quantity and calculate the total cost of them in order, as well as appropriate tax and shipping charges.

All this data will be stored on the UTS 400 diskette subsystem to support other portions of the transaction.

At the completion of the transaction, this information will be stored on the diskette subsystem and will be used to periodically produce printed outputs to be used by warehouse personnel and shipping clerks in the picking, packing, and shipping of the order to the customer.