HARDWARE ASSISTED POLLING FOR
AN ONLINE SYSTEM

Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Electrical Engineering.

by

Federico Solorzano Brauer

The School of Engineering
UNIVERSITY OF DAYTON
Dayton, Ohio
November 20, 1974

Approved

Chairman, Advisory Committee.

Director, Graduate Programs.

Dean, School of Engineering.
ABSTRACT

This Thesis will deal with the hardware changes that can be made to the National Cash Register online Multiplexer, C-621-103, part of the Century System, in order to improve the efficiency of the batch processing program working with the online real time financial system in a multiprogramming system, as well as the improvement of the response time of the terminals.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page No</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. INTRODUCTION.</td>
<td>1</td>
</tr>
<tr>
<td>II. OPERATION OF THE SYSTEM.</td>
<td>12</td>
</tr>
<tr>
<td>Poll Numbers.</td>
<td>14</td>
</tr>
<tr>
<td>Transmission of Data.</td>
<td>14</td>
</tr>
<tr>
<td>Selection of the Adapter.</td>
<td>17</td>
</tr>
<tr>
<td>Send Operation.</td>
<td>21</td>
</tr>
<tr>
<td>Receive Operation.</td>
<td>26</td>
</tr>
<tr>
<td>III. SOFTWARE PHILOSOPHY.</td>
<td>30</td>
</tr>
<tr>
<td>Central Online Software.</td>
<td>34</td>
</tr>
<tr>
<td>Operating Systems.</td>
<td>35</td>
</tr>
<tr>
<td>Monitor.</td>
<td>37</td>
</tr>
<tr>
<td>I/O Executive.</td>
<td>38</td>
</tr>
<tr>
<td>Extremity.</td>
<td>39</td>
</tr>
<tr>
<td>Verify.</td>
<td>41</td>
</tr>
<tr>
<td>Disc Management.</td>
<td>41</td>
</tr>
<tr>
<td>Log Routine.</td>
<td>41</td>
</tr>
<tr>
<td>Display Routines.</td>
<td>42</td>
</tr>
<tr>
<td>Online Communications Driver</td>
<td>42</td>
</tr>
<tr>
<td>Input Cycle.</td>
<td>45</td>
</tr>
<tr>
<td>Output Cycle.</td>
<td>48</td>
</tr>
<tr>
<td>IV. A PROPOSAL FOR IMPROVING THE EFFICIENCY OF THE SYSTEM</td>
<td>52</td>
</tr>
<tr>
<td>Poll Table Termination.</td>
<td>59</td>
</tr>
<tr>
<td>Message Termination.</td>
<td>59</td>
</tr>
<tr>
<td>Presentation of the Problem</td>
<td>61</td>
</tr>
<tr>
<td>Polling Adapter Operation.</td>
<td>64</td>
</tr>
<tr>
<td>Polling Adapter Components.</td>
<td>67</td>
</tr>
<tr>
<td>Selection Logic.</td>
<td>68</td>
</tr>
<tr>
<td>RAM Logic.</td>
<td>70</td>
</tr>
<tr>
<td>Input/Output Transfer Logic.</td>
<td>70</td>
</tr>
<tr>
<td>Termination Logic.</td>
<td>70</td>
</tr>
<tr>
<td>Poll Word Control Logic.</td>
<td>73</td>
</tr>
<tr>
<td>Non-Variable Poll Word Characters Logic.</td>
<td>73</td>
</tr>
<tr>
<td>Jumper Options.</td>
<td>73</td>
</tr>
<tr>
<td>RAM Matrix Addressing Logic.</td>
<td>73</td>
</tr>
<tr>
<td>Project Operation.</td>
<td>73</td>
</tr>
<tr>
<td>V. CONCLUSIONS.</td>
<td>80</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>81</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE No</th>
<th>PAGE No</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>Two wire Line Mode</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>Four Wire Line Mode</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>Point to Point System</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>Multipoint System</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>Components of the Multiplexer</td>
</tr>
<tr>
<td>6</td>
<td>13</td>
<td>Online System</td>
</tr>
<tr>
<td>7</td>
<td>16</td>
<td>Asynchronous Adapter</td>
</tr>
<tr>
<td>8</td>
<td>22</td>
<td>Send Operation</td>
</tr>
<tr>
<td>9</td>
<td>27</td>
<td>Receive Operation</td>
</tr>
<tr>
<td>10</td>
<td>31</td>
<td>Foreground/Background Memory Requirements</td>
</tr>
<tr>
<td>11</td>
<td>31</td>
<td>Sharing of Processing Time</td>
</tr>
<tr>
<td>12</td>
<td>40</td>
<td>B l Memory Map</td>
</tr>
<tr>
<td>13</td>
<td>43</td>
<td>Operating System</td>
</tr>
<tr>
<td>14</td>
<td>44</td>
<td>Operating System (Cont.)</td>
</tr>
<tr>
<td>15</td>
<td>46</td>
<td>I/O Operation</td>
</tr>
<tr>
<td>16</td>
<td>47</td>
<td>Data Transfer</td>
</tr>
<tr>
<td>17</td>
<td>50</td>
<td>Communication Tables</td>
</tr>
<tr>
<td>18</td>
<td>50</td>
<td>Communication Tables</td>
</tr>
<tr>
<td>19</td>
<td>53</td>
<td>Financial and Retail Software</td>
</tr>
<tr>
<td>20</td>
<td>56</td>
<td>User / Executive</td>
</tr>
<tr>
<td>21</td>
<td>57</td>
<td>OCD Routines</td>
</tr>
<tr>
<td>22</td>
<td>58</td>
<td>OCD React Routines</td>
</tr>
<tr>
<td>FIGURE No</td>
<td>Description</td>
<td>PAGE No</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>23</td>
<td>Polling Timing</td>
<td>63</td>
</tr>
<tr>
<td>24</td>
<td>Poll Operation</td>
<td>66</td>
</tr>
<tr>
<td>25</td>
<td>Polling Adapter Flow</td>
<td>69</td>
</tr>
<tr>
<td>26</td>
<td>Selection Block Flow</td>
<td>71</td>
</tr>
<tr>
<td>27</td>
<td>RAM Loading Block Flow</td>
<td>72</td>
</tr>
<tr>
<td>28A</td>
<td>Polling Adapter Logic</td>
<td>74</td>
</tr>
<tr>
<td>28B</td>
<td>Polling Adapter Logic</td>
<td>75</td>
</tr>
<tr>
<td>29</td>
<td>Polling Adapter Logic</td>
<td>76</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

A computer is considered to be online when a remote device connected to that computer and the computer itself interact in such a way that each affects the processing operation of the other. If the exchange of information between the external device, the terminal, and the computer is sufficiently rapid so that the computational results have an immediate and desired effect upon the external system, of which the device is part, the system is said to be operating in real time.

Of key importance in determining whether real time operation is possible, is the system response time, defined as the time required by the system to react to and to respond to an external input.

If a number of devices are connected to a single computer and if each of these devices is capable of communicating with this computer, a multiple access system is formed. If each of these devices can be serviced either sequentially or simultaneously by the computer and the user is given the impression that the whole processing system is servicing him only, the online system is termed a time shared system.

Time sharing activities center around the more efficient use of a single console digital computer system. In particular, the input or output of information
usually requires considerably more time than does the performance of a long sequence of arithmetic operations. For example, the input or output of a few characters may require milliseconds, whereas an addition or subtraction may be performed in a few microseconds; consequently, if only a single program is processed on a digital computer, the arithmetic unit must stand idle for a considerable percentage of time while input or output is taking place. To overcome this shortcoming, multiprogramming concepts were introduced. In this mode of operation several programs are stored in the computer memory and are processed "simultaneously" in an interleaved fashion, such that while one program makes use of the arithmetic unit another program employs the input-output equipment. This approach to improving the efficiency of computing system has been extensively applied in modern computers.

The term multiprogramming refers to the concurrent operation of two or more programs within a computer, whether output equipment is involved or not, such that one program need not be completed before another program is started. A multiprocessing system is one in which several programs or portions are executed simultaneously within a system consisting of two or more processing units.

Of key importance in a multiple access online system is the capability of the external devices to interrupt the program being processed by the computer so that access for input or output required by the device may be
given immediately. Usually whenever the external device desires access to the computer it transmitted a signal in the form of a changed logical level to a special input channel of the computer. The term interrupt is used to designate the capability of the computer to react to this signal by interrupting the normal data flow within the computer. Under these conditions the external devices may be processed by the central computer either in the order in which interrupt signals are received, or in an order determined by a priority schedule. In either case, if there is an appreciable volume of work, these external channels must wait on line until the computer is ready for them. The term queue is used to designate these "lines" which are formed by incoming messages. The queue problem becomes particularly complex if it is decided in advance that certain devices and certain messages have greater importance than others, and should therefore be handled first. Under this condition a priority interrupt system is used, and messages with higher priority are processed before messages with lower priority.

Each peripheral has assigned to it in memory a special area called a control word. This control word contains beginning and ending addresses for data for input and output operations for that peripheral. It also contains a special character which is used to indicate the status of the peripheral when an operation is terminated. The termination character is used to interrupt the main program
flow and causes the processor to enter routines to deal with the peripheral. Since the only indication that a peripheral has terminated is a status character in its control word, it becomes necessary for the program to scan all control words in order to locate the terminating peripheral.

Since 32 or 256 control words can be available according to the type of software in use, the control word scanning process can occupy a long period of time. If very few peripherals are connected that can cause termination then much time is wasted in searching control words where a termination status character can not exist. For this reason NCR online equipment employs a feature called HASQ: Hardware Assisted Software Queueing. This makes it possible for the control word address of the terminating peripheral to be "queue" in memory so that the program need only to see this queue to determine the terminating peripheral.

Generally with an online system, the processing unit is called "Central" and the remote devices are termed "Terminals".

The connection between central and terminal is normally made with two or four wires per telephone line (see Figure 1 and 2), input and output to the lines being made by a pair of modems (modulator-demodulator), which contain a transmitter and a receiver. The transmitter uses only two frequencies: 1200 Hertz and 2200 Hertz. The first is used when the transmitter is on but is not sending data or when it is sending a one bit. This frequency is named Mark con-
TWO WIRE LINE MODE

Figure 1
FOUR WIRE LINE MODE

Figure 2
dition. The second frequency is used to transmit a zero bit and is named Space condition.

If the system uses only two wires per telephone line, it can work in a half duplex operation, which means that either central or terminal can send data but not simultaneously. When central or a branch wants to transmit, it is necessary to wait until the other carrier goes down and the required carrier comes up and is stable. The online system uses four wires normally, to eliminate the "turn around time". In this way the system can work in full duplex; data can be sent and received at the same time.

It generally can be said that the operation of the online system can be divided into point to point and multipoint operation, as shown in the Figures 3 and 4. In the first case the system uses only one terminal per line. In the second case, it is possible to use more than one terminal per line.

At central the computer includes a memory and a processor that has eight lines of communications, or trunks. Each trunk has eight positions that are used to connect to peripherals. The computer uses a peripheral called a Multiplexer to send information to the terminals and, as the name indicates, it switches the data to and from different terminals, placed in different branches, through its adapters (Buffers), sending or receiving a character at the time.
POINT TO POINT SYSTEM

Adapters

Processor - Multiplexer - Telephone Lines - Terminals

CENTRAL

Figure 3
MULTIPOINT SYSTEM  (POLL SYSTEM)

Central

Terminals

Figure 4
The C-621-103 online multiplexer is capable of handling up to 126 asynchronous adapters with double control word, one interval timer, used to measure amounts of time, and one HASQ adapter (see Figure 5). Because there are so many adapters, there is a scanner that is used to give the opportunity to output or input a character in sequence to the adapter that requests this service. The scanner has two counters that are named unit and group scanner.

Normally after the first adapter selection only the unit scanner is free running.
COMPONENTS OF THE MULTIPLEXER

Figure 5
II. OPERATION OF THE SYSTEM

A block diagram of the online financial system is shown in Figure 6. The multiplexer has the function of selecting the adapter and the telephone lines to be used with a specific terminal.

The adapter works like a buffer to compensate for the difference of speed between the processor and the telephone lines. Basically the multiplexer is a device that is able to select the different adapters, or buffers, and gives them the opportunity to input or output a character every time that they request a service from the processor. To each adapter is attached a modem which is a device that changes the voltage level signals that come from the adapter into frequencies to be transmitted through the telephone lines or vice-versa.

Four telephone wires are connected to each modem, two for send and two for receive. Although the system works in half duplex, it uses four wires instead of two to eliminate the problem of turn-around time.

Bridges can be seen in the diagram at the terminals (Branches), which are used to match the impedance of the load (modems) to the 600 ohms impedance of the telephone lines.

An understanding of the system operation will be helped by the following explanation of the operation of
ONLINE SYSTEM

![Diagram of the online system showing the flow of data from the processor, through the C-621 multiplexer, adapters, modems, and branches #1 and #2, with connections to remote controllers, terminals, and telephone lines.](image-url)

Figure 6
the terminals.

**Poll numbers**

All the terminals in a branch are connected together in such a way that the information that comes to one terminal is presented to all the others. In order to differentiate between them each terminal uses a specific identification which is called a poll number. Each terminal can respond to its own poll number only and will totally ignore all others.

The way in which central distinguishes one terminal from another in the same branch, as was mentioned, is by using a poll number, which is a unique number assigned to each one of the terminals. Of course terminals that belong to different branches, and which therefore use different telephone lines as well as adapters, can use the same poll number. In this way the information sent from the processor will be presented to all the terminals in a branch or to terminal controllers if they are used, but only one will accept it.

**Transmission of Data**

The mode of transmission between central and terminal is asynchronous. This means that the code that is used is going to have two extra bits to synchronize each character with the terminal or the adapter. The code used by the system is the American Standard Code for In-
terchange of Information (ASCII) and it uses seven bits.

The adapter has three registers to store data which are called A, B and I/O (see Figure 7). The adapter's A register is used only for output operation while the B register is used only for input operation and the third register, I/O, is used for both.

This adapter works only in an asynchronous mode; it uses seven ASCII bits plus parity and two bits to control the transmission of the character: a start bit at the first position to initiate the synchronization of the character and a stop bit at the tenth bit position to signal the end of the character.

During input operation the start bit will initiate the creation of clocks which are going to sample the bits that come in the form of voltage levels, right in the middle of its duration in order to be sure to get the sample in the most stable part of the waveform; this information will be stored in the I/O register.

The start bit will also initiate a counter that counts the number of bits which are going to be stored in the buffer. When the stop bit appears the count will be ten and the counter, as well as the sampling clocks, will be reset and then wait for another character to be sampled. Meanwhile the character will pass in parallel to the B register, to wait for service from the processor in order to input the character to memory. The purpose of the stop bit was to mark the end of the char-
ASYNCHRONOUS ADAPTER

From Processor

A Reg.

I/O Register

Receive Data

Send Data

B Reg.

To Processor

Figure 7
acter as well as to give time to pass the character from the I/O register, which interfaces the adapter with the modem, to the B register that interfaces the processor.

As soon as this B register is full, the adapter will raise a request for service to the processor to input this character.

The adapter request will be serviced sequentially by the scanner in the multiplexer, so it is possible that the adapter may have to wait for service.

During output operation, as soon as the adapter detects that the buffer is empty it will raise a request for service to the processor, which will answer this request by sending a character which is stored in the A register. At the same time it will start a bit counter, a clock to sample the bits of the character which is sent to the terminal, as well as forcing the creation of the start bit. When the bit counter goes to the count of two the character has been transferred from the A register to the I/O register to be shifted out and the A register cleared.

At count ten the stop bit will be created. Because the character will have been transferred out of register A, at this time a request for a new character will be sent to the processor again.

Selection of the Adapter

The first step that must be carried out at central is to select the adapters for the operation of out-
put, because before a terminal can send data to central it must receive a group of characters from central asking if it has a message to send or not.

The processor, through the use of software, will prepare the different adapters to output the characters (Poll Word) that are going to create the so called "handshaking" between central and terminal.

In order to prepare for this operation it is necessary to select the different adapters for poll outward operation; the selection consists basically of sending from central three characters called "Peripheral Address Field" (PAF).

The first character will select the trunk and position but only the position will be sent to the multiplexer to be compared against a Berg connector that can be wired to a number that is convenient for the system. If the comparison is good then the multiplexer stores the result in a register and sends back a character to signal the processor that the trunk and position were accepted, but if the comparison for any reason failed, the hardware in the processor will store a status of inoperative and the operator will be notified.

The second PAF character will designate the number of the adapter, one out of 256 possibilities. The previous PAF character enables the reception and the storage of the number of the adapter into a register. In a similar way as with the first PAF there will be an answer back to
the processor in the form of a "character received" logic term, to acknowledge the reception.

As soon as the processor detects that the second character has been received, the software sends the third PAF character that will designate the function to be performed, which in this case is an output function, called poll outward. This last PAF character will start a sequence of events. When completed the right adapter will be selected for the right function. These events start with the comparison of the number of the adapter that was stored during the second PAF, against a Berg connector placed in the adapter, that was strapped previously to this number, and if the result is satisfactory, will enable the acceptance of the function that comes in this third PAF. This function is stored in the adapter and then sent back to the multiplexer to be compared with the function that was sent by the processor; if the result is good the multiplexer will send to the processor the logic term "character received" for the third time, as well as the two terms "end of control information", which means that the multiplexer does not need any more PAFs, and "trunk conditionally busy", which means that only the multiplexer can be selected again in case any other peripheral was connected to the same trunk. When the processor receives these terms it resets all of the logic terms that were used for selection and returns to the main program to execute the next command in sequence.
In the adapter, the combination of the logic terms that represent that the adapter was conditioned for an output function, and the condition that the buffer is empty, will enable the adapter logic to raise a signal called "request for service". In the multiplexer there is a scanner that is divided into unit scanner and group scanner. Like it was mentioned before the unit scanner is running all the time and the group scanner is initially reset to zero. As soon as the unit scanner matches with the adapter's unit Berg connector, the group scanner will start running; meanwhile the unit scanner is held in the position where it matched. This second scanner will run until it matches the group Berg connector; at this point in a similar manner as in the unit scanner the group scanner will be held in the position where the comparison is equal. As a consequence of this matching, the adapter will force the multiplexer to raise a signal to the processor called "channel request", together with a signal that will specify that this request for service is for an output operation.

The processor will now interrupt the batch processing to attend to this service. The configuration of the scanner will select a specific control word in which the software places the address of the field in memory that can be used by each one of the peripherals. This control word contains the first and the last addresses of the output field as well as the status of the adapters. This status character is very important because when one is input by
an adapter it will cause termination and therefore the processor will interrupt the batch processing to analyse what kind of status exists in order to execute the relevant software flow.

There are many status characters, but they can be divided basically between those that occur during selection and those that occur after selection.

In the first type can be: "inoperative" if the selection failed or "command initiate" if it was good. The second category can be divided into two kinds: the normal terminations and the termination by error.

**Send Operation: Data Transmission from Terminal to Central.**

As soon as the first adapter is selected for output operation it will ask for the processor to output the first character of the poll word (see Figure 3). If there is another selection in progress for another adapter, it has priority over input-output of data. However, if the request for service to input or output of data was raised before the next selection, the first has priority over the second.

The first character that is going to be output as part of the poll word is EOT, end of transmission. This character will reset the logic at the terminal if for any reason is not already reset, and place it in a condition to receive the poll word, as well as advance the terminal logic to enable it to receive the poll number. The second character is the poll number, which is unique for one and
Figure 8
only one controller in a group of branches that uses the
same adapter at central. This character as well is going to
advance the terminal logic, to prepare the terminal to re-
ceive the third and last character, the ENQ, enquire. This
will enquire if there is a message to be sent to central or
not. At this point the terminal takes the flow for sending,
if the teller depressed any "activator" key, or the flow for
sending automatically to central an EOT, if not.

Before we continue with this explanation let
us see what happened at central at the end of the output of
the poll word. In reality, the poll word is made up of four
characters and they are: EOT - POLL# - ENQ - DLM. We studied
the first three so, let us examine the purpose of the last
one. This last character is named delimiter and like its
name indicates, it is used to delimit the different poll words
in the memory of the processor. There will be as many poll
words as controllers or terminals connected to the line con-
trolled by the adapter.

The characters in memory will be output if and
only if the adapter so requests. So the question that arises
immediately is how it is that only four characters are going
to be output? Well the solution is simple. The adapter is
the device that controls how many characters will be sent
out and this output of data will continue as long as the adapt-
er is in output mode. The last character, called delimiter
(DLM), will change the adapter from output to input mode
automatically, destroying in the process this character
which has no function in the terminal.

Now that the adapter is prepared to accept data (input mode) the terminal can send a message or an EOT term which means no message. Let us analyse what happens when the terminal sends the EOT character to central. In order to avoid unnecessary interruptions to the processor the EOT will be used at central only to change the adapter back to the output mode and at the same time this EOT character will be destroyed, since it has no other purpose than to signal to central that the terminal has no message to send but that it is present in the system and that the communications is working well. A timer is used to verify that this EOT character or the message is received in a reasonable time.

As soon as the adapter is returned to output operation it will request the first character of the next poll word; in order to get the right character the software will increment the address in the control word with each request for service from the adapter, and in this way the correct address is always accessed. The operation continues in this form, sending out the four characters of the poll word and then changing the adapter from output to input mode, waiting to receive the message or EOT control character to change from input to output in order to access the next poll word.

If a message is received, it is checked for
horizontal and vertical parity and the results of this check will be returned to the terminal, which has been in input mode waiting for the result of this operation. The answer-back will be a single character ACK or NAK; ACK means positive acknowledge, and NAK negative acknowledge. The terminal will send EOT if the check was good in order to terminate the whole sequence. If NAK is received, the original data will be transmitted to central again. This retransmission can be carried out a maximum of three times, after which the terminal will be placed in an error condition to inform the operator.

The input message is framed by two control word characters which indicate the start and finish of data. These terms are called respectively start of text, STX, and end of text, ETX. The STX control character has the specific function of starting the sum check beginning with the next character in sequence as well as indicating that the next character is the first data character. In a similar way the other control character ETX is going to signal the end of the data as well as indicate that the next character is the sum check character, which terminates the input transmission. The sum check can be done by software or by hardware and if the latter, the BCC, as is known, will not be input to central.

At the end of every operation the software will request that the adapter input its status character in order to know what to do next; so after the BCC is input, if the sum has been done by hardware, the status that will be
input will reflect the result of this operation as well as the check of vertical parity.

When the message sent from the terminal is in memory, and if there were no errors, the software will send back ACK to end this "send operation". Now the message will be analysed by the program and as a result of this will send a message to the terminal to continue or finish the transaction started by the terminal.

**Receive Operation: Data Transmission from Central to Terminal.**

In order for central to send a message to the terminal it is necessary to establish a hand shaking routine again between the two system components, central and terminal.

The first step is initiated by central, which will select the adapter for output operation. As soon as the adapter is conditioned for output it will request service from central to send out the first character, which is an EOT, end of transmission. The second character is the select number, which is of similar form to the poll number. It is a unique number that distinguishes each controller from the others for output. The third character (see Figure 9), is another select number that distinguishes each terminal from the others in one controller. The fourth character is an enquire, ENQ, which enquires if the terminal is ready. The final character is a delimiter, DLM, which changes the adapter from output mode to input mode. This delimiter is destroyed because it has no function in the terminal.
Figure 9
At the branch all the remote controllers accept the EOT but only one is going to accept the rest of the characters so the other controllers will remain in idle condition.

The selected terminal after having received these three characters sends back the two select numbers followed by ACK, to signal that there were no errors during the transmission. These characters will be analysed by the software to verify that the terminal is the same one that sent the input message previously. The ACK will cause the adapter to input a termination character giving the status condition of the adapter and it will go to the idle condition.

The next step is the transfer of the message that contains the transaction which was done at central, with the data and control characters that form the output message. To do this it is necessary to select the adapter again for output mode. When the adapter detects that it is conditioned for output and that the buffer is empty, immediately it will raise a request for service to the processor, to get the first character of the message. Again the message will consist of the STX, start of text, data, ETX, end of text, and sum check character which can be created either by hardware or by software.

When the adapter detects the end of text character, it will give time to output another character, the BCC character, before it changes automatically from output mode to input mode. The terminal will analyse the message for vertical parity errors, bit number eight giving even parity
to the character, and BCC errors. After this the terminal will execute the contents of the message, which may or may not conclude the current transaction.

The final step of this operation of "Receive" will end when the terminal sends back to central a positive acknowledge, ACK, meaning that the message was received without errors and that the operation was executed at the terminal. If for any reason the message contains an error that justifies the sending back of a NAK, negative acknowledge, central will repeat the message up to three more times before the operation is aborted. This condition will be recognized by the terminal which will turn on the "Receive Error" light after the third retrial.

To acknowledge the reception at central of this character, it will send back to terminal a final EOT character, finishing the Receive Operation.
III. SOFTWARE PHILOSOPHY

The NCR B2 operating system is designed to permit dual programming; that is, the sharing of the processor between two programs resident in memory at the same time. This method of processing can be especially useful in online systems. It allows a normal batch program to take advantage of available processing time while the online program is idle awaiting input from remote terminals. When used in this manner (an online program and a batch program concurrently residing in memory), dual programming is referred to as online dual programming.

In online dual programming, memory is divided between an online program and a batch program. The online program referred to as the foreground program, resides in the high address portion of memory; the batch program, referred to as the background program, resides in the low address portion of memory. A standard B1 batch program, which follows all system conventions, may be run as a background program (Figure 10).

A minimum memory size of 32 K is required for online dual programming. Of this 16K is required for the operation of the background (Batch) program. Because the amount of memory available to the online program in this system would be insufficient for most online applications a 64K online dual programming system is normally used.
FOREGROUND/BACKGROUND MEMORY REQUIREMENTS

16K
Batch

Background Program Area

Online

64K

Foreground Program Area

Figure 10

FOREGROUND/BACKGROUND SHARING OF PROCESSING TIME

Processing Time

Foreground Program

All Tasks Suspended

Task Activated

Background Program

Figure 11
In dual programming, the foreground (online) program always has priority over the background (batch) program. However, when all tasks in the foreground program are suspended (awaiting input from remote terminals or awaiting completion of input-output operations), the batch program assumes control of the processor. The foreground program regains control when one of its tasks is activated (input received from a remote terminal or an input-output operation completed (Figure ii).

Operation under the dual programming environment begins with the loading of the online foreground program. One of the first functions that the online program performs is the calling of the B2 initializer. The initializer sets a flag in reserved memory to a one to indicate that dual programming is in processor. This flag remains set to one until deinitialization of the dual programming environment is to be initiated.

The online foreground program which was initially loaded into memory remains in memory throughout the entire online processing day (until B2 is deinitialized). During dual programming the online foreground program has priority, like was mentioned before, over any program residing in the background. The background is given control only when the online is idle.

Although background programs have a lower priority than the online foreground program, operation within the background is similar to operation under B1 operating
system (Batch program). When the background is first given control, Monitor is called into memory; Monitor is used only in the background under B2 dual programming.

Once in memory, Monitor requests the loading of the first background batch program. The background program loaded is given control and background operation continues until input from remote terminal is received or an input-output operation for online program is completed. When this occurs the background program is suspended and the online program receives control. The background remains suspended until the online program again becomes idle.

Whenever a background batch program executes a FINISH instruction Monitor is called into memory and requests the loading of the next background program. This type of Monitor controlled run-to-run linkage is similar to that of the B1 operating system. However, both the functions of Monitor and the background program are subject to suspension whenever the online program requires control.

Monitor will automatically call each subsequent background program from the disc or COT (Card reader or Tape reader) control string. However, if a COT control string is being used and the COT reader is empty at the time Monitor accesses it, an I/O writer message is displayed informing the operator of the situation. The operator should respond to the message immediately; this will free the I/O writer for possible use by the online program. Monitor will then continue to access the COT for the next control item with-
out redisplaying the wait message.

In dual programing the I/O writer is shared by both the foreground and background. All I/O writer messages originating in the foreground are preceded by an F, while those originating in the background are preceded by a B. Operator responses to I/O writer messages are not preceded by special character.

Operator responses should also be made as quickly as possible to free the I/O writer facility for other messages. It is extremely important that the I/O writer be kept free for possible foreground messages to avoid delaying the operation of the online program.

At the end of the online processing day, the operator will normally initiate an end-of-day routine within the online foreground program. The last function executed by this routine should be a FINISH instruction. This causes the reserved memory flag to be set to a two and the foreground permanently suspended. Once this instruction is executed in the background program, Monitor is called into memory. Monitor sensing the flag set to two, initiates deinitialization of the system. Deinitialization results in the E2 operating system being replaced by E1 operating system.

Central Online Software

In financial online systems, the customer records are kept at a central computer site, and these records are updated by the terminals located at the main branch
offices. Input from a terminal to central is permitted by central polling the terminals. Output from central to the terminals results from central selecting the remote terminals. The poll or select by central controls the flow of data, and the poll or select is controlled by the central computer program.

Central software is divided into three basic groups: user program, online communications driver (OCD), and Operating systems (OS).

The user program controls the customer records and the central file records. This program varies with each type of application.

The Online Communications Driver is a set of instructions that are tailored for specific types of remote devices.

The Operating System is a group of routines that interface the user program commands and the hardware controls and data transfer procedures.

Operating Systems

The Operating systems are a group of major routines (Monitor, Disc Management, etc.) made up of many minor routines (Data Traffic Controller, Dispatch and Interrupt service, etc.) to run user programs with a minimum of coding, memory space, and throughput time. Random or sequential file processing and the use of three different compilers (NEAT/3, FORTRAN, and COBOL) are permitted by the operating
The operating systems are not restricted by various upward hardware configurations, by the degree of simultaneity used, or by the number of input-output (I/O) buffers specified by the user.

The software of the operating system is modular in design (many interrelated parts), and resides both on disc and in memory. The majority of the software is disc-resident which saves valuable memory space. The memory-resident portion (Resident Executive) handles all normal operating functions such as setting up and executing all user requested I/O operations and the normal verification of compiled I/O operations. The disc resident portion is read into memory only as needed (when unusual operating conditions are encountered by the resident executive).

The dual disc unit lends itself ideally to the modular design of the operating system software. Any part of the disc resident portion of the software can be accessed randomly as needed. The two discs are designated as the current and the alternate system discs.

The current system disc contains all of the operating system software, plus the dynamic work storage area (an area containing such system information as the disc log and a list of all the peripherals available to the system) and the user's program and data files. The alternate system disc contains the same software as the current system disc, but not the current dynamic work area.
Either of the system discs can be replaced with another disc pack, provided that certain rules are followed. When the current system disc is changed, the log and the dynamic work area data are automatically copied to the alternate system disc, which then becomes the current system disc. The new disc pack becomes the alternate system disc.

When the alternate system disc is changed, no copy takes place unless the program currently running resides on that disc. If the program does reside on the alternate disc, program overlays are automatically copied to the current system disc. In this way, the operating system insures the latest version of the system software is maintained on all available system disc packs.

The system discs contain Monitor, the I/O Executive, Extremity, Verify, Disc Management, and the Log and Display Routines. These routines provide the necessary control when unusual operating conditions are encountered.

Monitor

Monitor is read into memory at the start of the day by the Monitor boot, and between programs by the resident portion of the operating system. At the start of the day, Monitor reads in (from the card or paper tape reader, COT) the date, the peripheral data, and the name of the first program to be run. Monitor uses this data to set up memory for a new processing day, to locate the desired program from disc, and to transfer control to the beginning of that
program.

Monitor also has the ability to run an entire series of programs with little or no operator intervention. The names of the programs to be run are set up in the desired order (called a control string) and stored on disc. The name of the control string is input to Monitor by the operator, through the card or paper tape reader. Monitor then runs each program in the sequence of the control string, providing run-to-run linkage between programs.

A program named in a control string can be run on a conditional basis, if desired, through the use of IF statements in the control string. When Monitor encounters an IF statement, it runs the next program named in the control string provided the IF condition is satisfied; for example, if today is Tuesday, run program A, etc.

I/O Executive

The I/O Executive is divided into two parts: the disc resident portion and the memory portion (the resident executive). The memory resident portion contains routines that perform such common program functions as processing input-output requests, calling system overlays, processing miscellaneous program and hardware errors, and supervising linking between subroutines within both the user program and the operating software.

At the start of the day, Monitor determines the system memory size and then maps the memory resident executive in one of two formats. One format applies to all systems with
memory greater than 64K. When larger memories are used, some of the more frequently used disc resident subroutines become memory resident, providing faster program execution (Figure 12).

Part 1 of the resident executive contains software routines, lists, and tables required to run any program. Part 3 which is used only on systems with memory sizes greater than 64K contains some of the more frequently used routines that must be called from disc when smaller memories are used. Part 2 is used to input disc resident routines as needed to handle relatively infrequent situations. Extremity and verify are read into this area.

**Extremity**

Extremity opens and closes the files in a user's program automatically, if the programmer desires. Before Extremity opens a file it verifies that this is the exact file requested, thereby preventing the premature destruction or improper use of a file.

File protection is provided by dates or generation numbers. The operating system software has access to a 3 year calendar, which enables the system to keep track of the day, the day of the week, the beginning of the work week, and other dating information for file protection.

Files can also be selected by generation numbers which Extremity uses to insure that the desired file is opened. First, second, and third generations of the same file,
Less Than 64K

Resident Executive (Part 1)
3.5K

User

Program

Resident Executive (Part 2)
.5K

Greater Than 64K

Resident Executive (Part 1)
3.5K

User

Program

Resident Executive (Part 2)
.5K

Resident Executive (Part 3)
1.2K

Figure 12
with the same name may exist.

Verify

The memory resident verify routines handle all normal I/O verification. However, if a condition such as read or write failure occurs, the disc resident Verify 2 routines are read into memory. These verify routines perform automatic retries of an I/O operation that failed. If the verify routines can not correct the condition, they cause other routines to be read into memory to inform the operator of the situation.

Disc Management

The Disc Management portion of the operator system software insures that no system information is lost during a pack change. Disc Management handles the automatic copying of the necessary information from the current to alternate system disc, when the disc pack changes are made. Disc Management also insures that the latest versions of software are copied to all the disc packs used in a system.

Log Routine

The Log Routines automatically maintain the system disc log, which contains information pertaining to the status of the system. Such information as hardware malfunctions or incorrect operating procedures by the operator is maintained in the disc log. The disc log can be printed to provide a permanent record of the entries made in the log during the day. A record of hardware failures is a
valuable aid in correcting hardware problems, just as a record of incorrect operating procedures is helpful to site personal in correcting operating problems.

Display Routines

The operating system Display Routines provide a communication link between the programmer and the operator, or between the operating system and the operator. Any time the operating system detects an abnormal condition (such as a programmer error), the display routines are read into memory to display a message through the console. From this display the operator is able to determine the action necessary to correct the abnormal condition.

The Figures 13 and 14 illustrate the interrelationship of the modules in the operating system, and the function each part performs. It does not show all the details of the operating system since it is intended only as an illustration of the basic concept of the system.

Online Communications Driver

The operating system uses a special set of routines that control remote communication terminals. These routines are called Online Communications Drivers (OCD) and they are tailored for specific types of terminals.

The OCD is responsible for all software functions relating to online I/O operations. Some of the software are as follows: verification of input and output messages, scheduling (priority) of input and output messages, and analyz-
The operator uses the console switches to read the 101 Boot into memory from cards or tape.

The 101 Boot reads the Monitor boot into memory from disc.

The Monitor boot reads in Monitor and the Resident Executive.

Monitor continues reading cards or tape to obtain the virtual date, actual date, and any peripheral data.

Monitor checks the operating system software version number on the current and alternate system discs. If both versions are not the same, Monitor stores a link in the Link List so that control may be returned to Monitor at a later time.

If the current system disc contains a later software version than the alternate, Monitor calls the Disc Management routines to copy the system software from the current to the alternate system disc. If the alternate has a later version, Monitor calls on the Log and Display routines to note this condition in the Log and display it to the operator for corrective action.

After the copy or the corrective action by the operator, control is returned to Monitor via the Link List.

Re-entry point.

Extremity searches the Disc Directory to find the named item and informs Monitor whether the item is a program or control string (assume program). Extremity passes the actual vector address of the control string to, and returns control to, Monitor.

Extremity searches the Disc Directory to find the named item and informs Monitor whether the item is a program or control string (assume program). Extremity passes the actual vector address of the program to, and returns control to, Monitor.

Extremity searches for and opens the necessary files (printer, cards, etc.). At each file is opened, Extremity calls the Log routine to record such historical data as the program name, the file name, and the date. If a file cannot be found Extremity calls the Display routine to inform the operator of the condition and options.

Extremity then calls on the Data Traffic Controller (a portion of the resident I/O Executive) to fill all but one of the buffers for each file. Data is therefore immediately available to Program A.

When finished, Extremity relinks to Monitor.

Monitor reads the remainder of Program A into memory over itself, and transfers control to the start of the program.

Figure 13
VERIFY makes extensive use of the memory-resident portion of the I/O interface, which contains all the software generally needed to handle common processing situations.

I/O transfers control to the memory-resident lists interface controller (LIC) to "retrieve" the input buffer areas, that is, the LIC maps the first record in the first buffer (filled during program load) available to the program and initiates the filling of the remaining buffer for that file. Control is returned to the user program before this buffer is filled.

The buffer is filled as the program simultaneously processes the file data from the first buffer. (When multiple buffers are used, the program is generally not halted while file data is input.)

When the buffer is full, the program is automatically interrupted at the completion of the current instruction and control is transferred to the memory-resident Verify 2 routine. This routine tests the input (I/O) operation for successful termination, and then either returns control to the user program or calls a disc-resident Verify routine to attempt error correction.

The disc-resident Verify 2 routine is generally successful in correcting such errors as bad reads from disc, transmission error, etc. Whether or not the error is corrected, Verify calls the Log routine.

If the error was corrected, the condition is logged and control is returned to the user program. If the error was not corrected, the condition is logged and the Display routine is called into memory.

The Log routine records data pertinent to the successful termination of the program, then transfers control back to the FINISH instruction, FINISH calls the Monitor back into memory, and the procedure begins again.

If the previous program was called from a control string Monitor looks to that same control string for its next instruction.

If the previous program call was from punched card or punched paper tape, or if the control string is finished, Monitor halts prior to looking at the reader for its next instruction.

Return to point A.

Figure 14

Page 44
ing problem conditions by use of diagnostic routines. To perform these functions, the OCD must interface to both the operating system and the user's program.

The interface to the operating system exists whenever an I/O initiation occurs. After being informed by the user program of an I/O initiation, the OCD presents certain I/O specifications to the Data Traffic Controller (DTC) routines of the operating system. These DTC routines actually perform the I/O execution as dictated by the peripheral address field (PAF) setups, and verify routines specified by the OCD (Figures 15 and 16).

Whenever a remote I/O termination occurs, the DTC uses special interrupt service routines to check the status characters, and then passes control to the OCD. The OCD verifies the message contents and logs results of the communications in tally tables. The number of tally tables kept by the OCD varies according to the number of remote units and the number of communication (telephone) lines in an online system. One table is kept for each remote terminal (unit tally table) and one is kept for each telephone line (link tally table).

The interface to the user program is the primary function of any OCD. As an example of this interface, an input and an output operation are briefly described.

**Input Cycle**

Input data from a remote terminal must be stored in a buffer area. Therefore, I/O buffer areas must be
Input/Output Operation

Figure 15
DATA TRANSFER (BLOCK DIAGRAM)

ANY COMMAND

REQUEST

EXIT TO OFF-LINE FLOW

RETURN TO INTERNAL FLOW

(CONT.)

SELECTED PERIPHERAL

UNIT RESPONSE NUMBER

CONTROL

COMPUTE DATA ADDRESS

TRANSFER DATA (1 CHAR)

N16 (OFF-LINE) DATA IN OUT

Figure 16
provided by the user program and these buffer addresses presented to the OCD. The OCD takes these addresses (beginning and ending) and loads them into the correct control word. The OCD then uses the operating system interface to cause a unit to be polled for input. As the terminal inputs a message, the multiplexer uses the control word to store the message at the correct location.

When the last character is stored, an I/O interrupt uses the operating system interface to place the OCD in control. The OCD verifies the message as valid or not valid. If valid, the OCD stores a code in the I/O buffer area and places the buffer address on a queue list. When the user program regains control, it accesses the queue list and processes the message pointed at by the top address on the queue. When an input message is processed, it usually results in an output message to the terminal.

**Output Cycle**

The output message is constructed by the user program and placed in an I/O buffer. The user program then executes a commend which puts the OCD in control. The OCD determines what terminal is to be selected and what telephone line (link) is used and then the proper link is checked for being busy. If the link is busy, the output operation is queued to a table which is accessed when the link is not busy. If the link is not busy, the OCD locates the control word to be used and inserts the I/O buffer address. Then the operating system interface is used to initiate the output
The comfile table points to the remote I/O routines.

The comlink table points to the comfile, comline and comunit tables used on the comlink.

The comline table points to the associated comlink table.

The comunit table points to the associated comfile and comlink tables.

Figure 17

One comfile table (normally) for each type of terminal. However, like terminals may be divided into several comfile tables if desired. Under no circumstances may unlike terminals be assigned to the same comfile table.

One comlink table for each adapter.

One comline table for each communication port used by the adapter.

One comunit table for each remote terminal in the online system.

Figure 18
of the message.

Since the output operation may not be completed quickly, the user must protect the output message in the I/O buffer. This protection must be maintained until the output is received at the remote terminal and properly acknowledged to central. Again using the operating system interface, the OCD is notified of the I/O termination which enables it to pass a code to the user program. This code causes the user program to release the protection of the I/O buffer area. The codes referred to are actually software status characters which function similarly to the hardware status character. The software status characters are referred to as SC1 and SC2.

The SC1 character reflects the type of instruction that the OCD is to perform for the I/O operation (input or output). An intermediate SC2 character is stored which indicates whether or not the OCD has accepted the instruction as valid. The terminating SC2 character indicates the terminating status of an I/O instruction.

In addition to the tally tables mentioned earlier, the OCD makes use of other types of tables: COMPILE, COMLINK, COMLINE, COMUNIT and POLL. These tables are used by the OCD and user program to control all I/O operations. Each table is defined and built from information furnished during program compilation. The information furnished varies for each online system configuration. An example of the table relationship to the hardware components is illustrated in Figure 17.
Although each of the communications tables is defined separately, all of the tables interact with each other to provide the operating system, the OCD, and the user program with an effective method of remote I/O communication. Figure 18 depicts the relationship between the different communication tables.

The OCD's are sets of routines that control the setup of commands and all related software functions for remote terminals. They interface to the operating system and to the user programs. The OCD's also use tables which are created at program compilation and create tally tables which are useful to the Technical Service Representative.
IV. A PROPOSAL FOR IMPROVING THE EFFICIENCY OF THE SYSTEM

In order to easily visualize some of the problems that exist in the software of a poll system it will be convenient to use a simplified block flow of the Financial and Retail software. In figure 19 can be seen the big picture of the software. It is divided into 3 parts: User program, Online Communication Driver (OCD), and I/O Executive (Operating System). It is not shown in the picture, but in the OCD there are 2 kinds of tables that need to be explained because the software will inquire continuously to find out which is the next operation that must be executed: polling or the output of a message. The first one is called Poll unit table and there is one table per link; the other is the Unit table and there must be one per terminal.

Inside of these tables, Poll and Unit, the software will store two codes that help the program to know what operation is next; one code is called ACT code and the other REACT code.

The operation of the software is as follows:
The B2 Executive is loaded into memory, then user files will be opened and control is passed to the OCD (Caller).

The OCD Caller will control the type of operation Poll (Input) or Output, and check to see if the link is in use by another terminal. The first time that the link is checked it should not be busy and then control will pass
FINANCIAL AND RETAIL SOFTWARE

Load B2 Executive

Open files

Caller: Poll the terminals or output a message.

link active

Scheduler: select operation

User process message

Interrupt Service routine (ISR): check status.

Data Traffic Controller (DTC): Initiate I/O Selection.

Poll: set poll table.

Select: set up output message.

Input message was received. Output ACK

Rotate Tables.

 ONLINE COMMUNICATION DRIVER (OCD)

Figure 19
to the Scheduler to prepare to poll table or the message to be output. The Scheduler will rotate the tables to see the ACT code of each one; if the code is zero the tables are rotate again to see the ACT code of the next in sequence. If the code is 3 the program will poll the terminals and finally if the code is 18 or 15 the program executes the output of a message to the terminals to which that unit table belongs.

Depending of the operation, polling or the output of a message, the program will pass control to the Executive (DTC) to select the multiplexer for one operation or the other. At the end of this operation the software passes control to the user program which will wait on a termination of either function, input or output. When the program is in the user, control can pass to process the online message if there is one or control can pass to the background program to process the batch program.

When one termination occurs, control is passed to the Executive (Interrupt Service Routine, ISR) to check the status and then control passes to the OCD to see which is the next operation in sequence for that link, by looking into the tables. If a message is received, the user program will take care of processing it, and prepare the answerback message.

After this brief introduction let us pass to study a little bit more of detail that is necessary in order to expose one problem in the online software.
(see figures 20, 21, 22).

After loading in memory the batch Executive (Bl), Monitor reads the next program in sequence that must be executed. If it is an online program, then the B2 Executive is loaded in memory and then the user program open files and request the OCD (Caller) to start polling the terminals, then places the poll unit table at the end of a queue list.

The Caller puts an ACT code 3 in the poll unit table, and checks if is active (input or output of data in that link); because this is the first operation for this link, this one obviously is not active, so the program passes control to the OCD Scheduler.

The OCD Scheduler will rotate the tables as long as they contains an ACT code of zero. The purpose of this rotation is to give the same opportunity to all terminals in the link to receive an output message. Eventually the program will find an ACT code of 3 in the poll unit table, and then control will be passed to the poll routine where the poll table will be prepared with the different poll words, one for each one of the terminals; but not only this, the PAF (Peripheral Address Field) will be prepared in order to select the multiplexer as well as the adapter that is going to be used to output the poll words. At the end of this operation the program goes to the Executive (Data Traffic Controller) to initiate the selection of the multiplexer and the adapter, and then will
Begin Operation

Load B1

Monitor call next program

MF set

Yes

Load B2

Executive

User open Files

User begin polling (input)

OCD Caller

I/O Termination

Queue Empty

Process one Command

I/O Termination

Output Ready

Output Message

OCD Caller (Branch to user on relink)

Data Traffic Controller (DTC)

Initiate I/O Selection

User (Relink)

Interrupt Service Routine (ISR)

I/O Verification

OCD React Routine

Figure 20
OCD ROUTINES

Caller

Input

Queue

Poll unit

Table

Poll unit

ACT 03

Yes

Link active?

No

User

OCD Scheduler

Scheduler

Rotate one unit Table

Yes

ACT 00

No

No

ACT 03

No

ACT 12

OCD Poll

No

ACT 15

No

ACT 18

OCD Select

Others

Select

Yes

More segments?

No

Sched.

Unit Table

ACT 15

Set up to output select word, message and EOT

Set up codes

FSC 06

REACT 00

B 2 DTC

Poll

Set up Poll Table

Set up codes

FSC 09

REACT 00

B 2 DTC

Figure 21
OCD REACT ROUTINES

Input Termination

React 00

Queue input to user

ACT 12

Set ACK in Buffer

Set up codes
PSC 03
REACT 09
B2 DTC

React 09

Yes
EOT

No
Resend ACK

OCD Scheduler

Output Termination

React 03

No
Last segment output

Yes

OCD Sched.

ACT 00

OCD Scheduler

Rebuild Poll Table

React 00

Remove units that timed out

Re-instate units sched. for retry

OCD Scheduler

Figure 22
relink to the user program.

The user program will remain suspended, waiting for a termination. Meanwhile the program can process online messages or pass the control to the batch program until a termination occurs.

In the online program two terminations can happen basically: 1. Termination because of the end of the poll table, and 2. Termination of the output or input message.

Poll Table Termination

As soon as a termination occurs the program goes back to the I/O Executive (Interrupt Service Routine, ISR), to check the status and at the end of this verification the program goes to the OCD REACT routine (00), output mode. The REACT routine rebuilds the poll table and takes from .2 to .5 seconds to do so. During this time the processing of data has been totally stopped.

From here the program goes to the Scheduler to rotate tables looking for a message to output, or to poll again.

Message Termination

When a terminal sends a message in response to a poll word, it will be stored in memory and at the end it will cause termination. With this termination the program passes control to the I/O Executive (ISR) to check the status, and then passes to the OCD React routine 00,
where the program queues the message to be processed later by the user program. Here the program changes, in the unit table, the ACT code to 12, and the REACT code to 9. Now the program prepares the response to this message, positive acknowledge (ACK), and the Peripheral Address Field (PAF) to select the multiplexer and the adapter to output this response. At the end of this operation the program goes back to the I/O Executive (DTC) to do the selection of the hardware. After this the program relinks to the user program where the input message will be processed, and the response message (output message) will be prepared. Meanwhile the user waits for the termination of the input message.

When the termination for the input operation ends, the program goes to the I/O Executive (ISR) to check for the status and then passes control to the OCD REACT routine 9, where the proper reception of the EOT that came from the terminal is checked. From here the program goes back to the OCD Scheduler to rotate the tables looking to output a message or poll the link again. Let us imagine that poll operation will be repeated so the sequence of events will be the same that was described before. At the end the program will go to the user to be suspended waiting for a termination. If the output message meanwhile has been prepared and the poll operation continues to be done by hardware, control is passed to the OCD Caller to place in the right unit table an ACT code of 18, then the program goes to see if the link is active and it will be if the
polling still is in progress if the link is active the program will go back to the user program to wait for termination. When it comes, like it was described previously, the program eventually will rotate and see the ACT code of 18 and the unit table that belong to the terminal that is waiting for the answerback message. The program passes control to the Select routine which prepares the select word, message, and EOT as well as changes the ACT code to 15 and the REACT code to 03.

From the OCD Select routine the program goes to the I/O Executive (DTC) to prepare the multiplexer as well as the adapter to output all the characters mentioned before. At the end of this operation the program goes, as usual, to the program of the user to wait for a termination. At the end of the receive operation the user program is interrupted for a termination from this operation and control is passed to the I/O Executive (ISR) to check the status. After this is done the program goes to the OCD REACT routine 03 to check if this is the last output segment; if it is then the ACT code is changed back to 00 and control is passed to the Scheduler to rotate the tables to see what is going to be done next.

Presentation of the problem

The main problem of wasting of time that can be seen in the previous explanation is that of the re-establishment of the poll tables, because if we imagine that a customer works 10 hours a day and he has a poll table that
contains ten poll words that need to be set again after every 30 msec and that this process consumes, like was mentioned before, from .2 to .5 seconds, then in a period of ten hours he will use from 2.5 to 6.27 hours (Figure23). Of this time at least half is negative acknowledged, and from the view point of software this time is totally lost for the processing of data.

The creation of a true HASP adapter (Hardware Assisted Software for Polling operation) that uses hardware instead of software to poll the terminals, can result in a 15 to 30 per cent saving of the time used during the run of an online multiprogram and at the same time, the response time of the terminal can be improved. But not only this, the use of the hardware poll tables, mentioned above, for test purposes, will enable the technician on many occasions to do troubleshooting without using the central processor to test the input handshaking; this means that there is no need to use the customer's batch processing time in the case of a failure of the online system.

The logic design of a HASP adapter must fill the following requirements:

First, be as simple as possible in order that the device can be adapted in the field to the machines that are already in the field.

Second, must use the same integrated circuit packs, if possible, that are now being used in the machine in order not to increase the number of spare parts required.
POLLING TIMING

Work time : 10 hours (36000 Seconds)
Input of Status : Every 300 msec.
Number of Requests to input Status : 36,000/.3 - 45,000
Time to re-establish the Poll Table : from .2 to .5 Seconds
Total time : From 2.5 to 6.27 Hours

Figure 23
for the correct maintenance of the equipment in the field.

**Polling Adapter Operation**

Let us suppose that the multiplexer has two adapters: one an asynchronous adapter and the other the Polling adapter with each one in a different scan position. The polling adapter should be prepared to be able to work in conjunction with the asynchronous adapter by using the proper strapping.

By using software the computer's operator can load the poll numbers in the polling adapter, using one poll number for each one of the terminals that are in the link in which is placed the asynchronous adapter. These poll numbers are stored in a RAM (Random Access Memory) chip, that is able to store up to sixteen characters for fifteen different links, by using first a select operation that is divided like it was explained in a previous chapter, in three steps that selects in the first one, the trunk and the position of the multiplexer. In the second the scan position of the polling adapter is selected and in the third step the adapter for the output function is selected.

The selection enables the transfer of data (Poll numbers), from the processor to the polling adapter; when the capacity of the RAM is filled, then the data transfer is called to an end.

Because the rest of the characters of the poll word are non-variable characters (EOT-ENQ-DLM), they will be created by hardware inside of the polling adapter,
therefore avoiding the need of storing these three characters for each one of the links; instead one set will be used for all the links.

In order to start the polling of the terminals the computer's operator (see figure 24) must select the asynchronous adapter, by software, to specific function of poll outward, that will put this adapter in the output mode. Then the adapter will request service to output the first character of the poll word, EOT, but the polling adapter inhibits the request to the processor and instead answers the service by sending to the asynchronous adapter this first character to be output to the terminal through the telephone lines.

As soon as the buffer of the asynchronous adapter is empty it requests service to output the first poll number that was stored in the RAM. Again the request to the processor is inhibited and answered by the polling adapter; at the end of this operation the hardware leaves prepared to be sent, with the next poll word, the next poll number in sequence that was stored.

The next time that the asynchronous adapter requests service, for the third time, the request is inhibited again and as before, answered by the polling adapter that sends the ENQ character to the terminals to ask for a message or an EOT if there is none. From the viewpoint of the terminal this is the last character of the poll word; however it is not the last for the asynchronous adapter that
POLL OPERATION

Processor selects an adapter for Poll outward function.

Adapter stores the function and goes to output mode.

Request for service to the processor

Buffer empty: request for service to output one character.

Request for service to polling adapter.

Buffer full: output one poll word charc. to terminal.

DLM?

Change to input mode and wait to receive the message or EOT.

Request for service to processor to input the message.

Figure 24
requests one more in order to change from output mode to input mode to receive the answerback from the terminal, without interrupting the processor to reselect the adapter for input mode.

The last time that the asynchronous adapter requests service to get the DLM character, it will, as on previous occasions, receive service from the polling adapter. When the delimiter character is detected, the asynchronous adapter changes to input mode. If an EOT is sent back from the terminal then the next poll word is sent in a way similar to that described previously. If a message was sent then it is stored in memory to be processed, and at the end of the Receive Operation the software will select the asynchronous adapter for the poll outward function, to continue polling the other terminals.

The hardware in the polling adapter is prepared in such a way, that at the end of the poll table will be a termination only if a terminal inputs a message during the poll operation. The purpose of this status is to give an opportunity to the software to output a message to the terminals during a Receive operation. If all the output messages are sent, the operation of termination at the end of the poll table is inhibited and is not repeated unless, like it was mentioned before, there has been an input of a message operation, or a previous one has not been answered yet.

Polling Adapter Components
The Polling adapter can be divided into the following components:

1. Selection logic.
2. RAM (Random Access Memory) logic.
3. Input/Output (I/O) transfer logic.
4. Termination logic.
5. Poll word control logic.
6. Non-variable poll word characters logic.
8. RAM matrix addressing logic

See Figure 25 for a block diagram of these components.

1. Selection Logic

The purpose of the selection logic is to enable the loading of the RAM matrix where the variable poll numbers are stored. To do this, it is necessary to use four peripheral address fields.

The first one contains the trunk and the position of the multiplexer in the system of the computer.

The second PAF contains the scan position of the polling adapter.

The third PAF can be the function, output, or the number of the RAM that has been selected to store during this operation; to be one or the other depends on the number of the RAM, whether the bit number seven is used or not. If it uses this bit, it is a must to send this character in the third PAF as an escape function; this means that bit eight must also be used.
POLLING ADAPTER BLOCK FLOW

Figure 25
The escape function is used because it is not necessary to validate the reception of the function inside the multiplexer.

The fourth PAF can be either the function or the RAM number, depending on the state of bit seven. The multiplexer interprets the bit seven as a signal to request more PAF (See figure 26).

2. RAM Logic

The RAM matrix consist of thirty two individual RAMs that are placed in pairs in order to store eight bits in each address; the result is that the matrix can store sixteen poll numbers for sixteen adapters or if needed eight poll numbers, if double ID numbers are being used.

3. Input/Output Transfer Logic

During selection the RAM number is stored in a register because it will be used to select one pair of RAMs out of thirty two that are in the matrix; and will also help select the appropriate address for the selected RAM. The I/O logic will request service from the multiplexer to load the RAMS with the poll numbers (see figure 27).

4. Termination Logic

The purpose of this logic is to signal to the processor that the maximum capacity of a RAM has been reached and that it is time to make the decision to continue loading RAMs or select the other adapters for the poll outward function to start polling the terminals.
Multiplexer in Idle mode. Clear poll word control RAM.

First P.A.F.: trunk and position.


Third P.A.F.: adapter position is presented as well as the function or RAM number.

bit 3 on? No

output function is stored (42)

Yes

RAM number is stored (40–7f)

Fourth P.A.F.: store RAM # (00–3F)


Processor drops selection terms. The Polling adapter request for service to load the RAMS.

Figure 26
During selection set the Polling Adapter for output mode and select a RAM to store the poll table.

Request from the processor the poll number that will be stored in the RAM.

Store the poll number in the RAM.

Table full?

No

Yes

Request for service to the processor to input status (table full).

Figure 27
5. Poll Word Control Logic

This logic outputs the different characters, from the poll word, in the right sequence, and at the proper time.

6. Non-Variable Poll Word Characters Logic

In order to cut production cost, logic common to all poll words will be used for the selection of the terminals.

7. Jumper Options

Because there are basically, in NCR, two kind of terminals, those that use four characters: EOT - POLL# - ENQ - DLM and the other that use five: EOT - CUA (Controller Unit Address) - DUA (Display Unit Address) - DC1 (Read) - DLM, it has been decided to put a group of option jumpers that enable the polling adapter to work with both kind of terminals in the same system but not in the same branch.

3. RAM Matrix Addressing Logic

The addressing logic as well as the enable of the RAMs depends on the scan position being used as well as the jumper options that give the opportunity to choose any scan position (from 00 to FF) to use this feature.

Project Operation

During selection, the polling adapter receives (see Figures 23 and 29) through the unit and group selection lines (2UD and 2GS) the scan position of the adapter that is going to be selected.
Figure 23 A
POLLING ADAPTER LOGIC

Figure 23 B
Figure 29
The scan position number is compared against a Berg connector, that is wired with the most convenient position to place this adapter in the system.

At the output of the comparators, the result is put in an AND gate with the clock, that has a frequency of 1 MHz. This signal will strobe the output function (2OPS), the kind of control word that is going to be used (2CW), odd or even, as well as the number of the RAM that will be loaded in the I/O operation that starts as soon as the selection operation is over.

When the function is stored, the polling adapter sends back the line 3AMR2 to be compared against the function that the multiplexer sent. If this comparison is successful the selection is over and the request for service flip-flop (RFS) is set.

As soon as the processor drops all the selection logic terms the scanner is free to run and the next time it addresses this adapter, the lines 3UAB and 3RQT are raised stopping the unit and group scanner and presenting a request of a character to the processor.

The processor will answer back with the logic signals JDRQR*, request received and HDAT1-9* lines, that contains the character that is going to be stored in the RAM; the RAM number selected was stored in the R register during selection, and will be used not only to enable one pair of RAMs but will select the address inside of the RAM.
With the last character the logic term JDTRM*, request for termination, will be sent to set the termination flip-flop; once set, the termination flip-flop forces the multiplexer to input a status and puts the adapter in an idle condition.

In this condition the enable and selection of the RAMs depends entirely on the scanner as well as the function and mode of the adapter that is currently being used.

It is important to note that the logic that controls the sequence of the poll words is reset to zero before any selection is attempted. This is accomplished by the use of the initialize (INT) logic term that systematically resets all sixteen addresses of the poll word control RAM; this forces EOT, end of transmission character, to be the first sent in each of those sixteen adapters that are using this feature.

At the same time this RAM logic enables the output of the poll word at the right time and in the same order that the programmer stored them in the RAM matrix.

When an asynchronous adapter is selected for the poll function and is in output mode, it will initiate a request for service as soon as it detects that its buffer is empty; if the strapping of the polling adapter is prepared to work for the scan position of the adapter that requests service, this request to the processor will be inhibited and the service is given by the polling adapter.
At the same time the polling adapter enables the logic in the multiplexer for the detection of control characters (DLM), disables the output data lines (HDATl-9*), and presents the character that will be output.

The scanner enables: First, the poll word control RAM, which enables the output of the poll word characters in the right order; Second, the selection of the RAM that contains the appropriate poll numbers for the specific adapter that requests the service, and selects one specific poll number inside of the RAM.

At the end of the service the RAM address for the poll word is incremented, to present at the next time, the next character in sequence.

Only when the poll character is sent, the address to access the next poll number is incremented, otherwise it remains the same.
V. CONCLUSIONS

The present online real time system is a good system that has been improved through the use of concentrators, terminal multiplexers and other refinements like HASQ (Hardware Assisted Software for Queueing Operation), but I firmly believe that it could be improved even more if the change proposed here could be incorporated. It would save the customer time and money since this change would take over the job of polling the terminals, leaving the processor free to run customer programs. The processor would be interrupted only when a terminal has data to send.

This would mean more production time plus saving 15 to 30% on compute time.

Another advantage of this change would be seen when the online system fails. It could be used by the technician for testing the equipment without the use of the processor. Thus the customer would be receiving maximum efficiency from the equipment with a minimum of expense.

A prototype was built to test the logic design of the polling operation. After a few refinements, it was working quite satisfactory through the use of Diagnostic Routines.
REFERENCES

Financial Systems. Field Service and Training Manual. (NCR MS 934)


621 - 103 Multiplexer Logic. NCR No 315-0523305, 315-0523853, 315-0523353, 315-0523841, 315-0523347

692 - 600 Asynchronous Adapter Logic. NCR No. 315-0523236.

621 - 103 Communications Multiplexer. NCR MS 553.

438-6 Window Machine Remote Cotroller. NCR MS749