bcc	title: LOCAL ECHOING IN THE COMMUNICATIONS SYSTEM	prefix/class-number.revision: OHWOW/W-41
checked: Larry L Barnes	authors: Paul Heckel	approval date: 12/31/69 / revision date:
checked: Duffie W Douglas		classification: Working Paper
approved: Mel		distribution: Company Private / pages: 9

ABSTRACT and CONTENTS

Local Echoing is of sufficient interest, subtlety, complexity, and orneriness to deserve a document of its own. This is that document. It discusses a procedure for resuming local echoing in the Remote Concentrator which assures all characters are echoed properly. It also describes procedures for outputting to a device that is in local echo mode.

INTRODUCTION

In a full duplex communications system, characters that a user types into the computer must be echoed (typed out). If this is done by the Remote Concentrator in the obvious way, many of the reasons for having full duplex mode are nullified. If it is done by the CHIO, the response is too slow. For this reason Local Echo Mode has been invented. The Remote Concentrator echos characters until a break character is typed, then it stops echoing and the CHIO starts echoing. When input is again needed, the Remote Concentrator resumes echoing.

Two problems present themselves. One is resumption of local echoing. The second is the stopping of local echoing by the CPU. Each is now considered in detail.

Local Echo Resumption

When a break character is input from a teletype, the Remote Concentrator stops echoing the characters as they are typed in. Eventually the CHIO will want to ask the Remote Concentrator to resume the local echoing. Unfortunately, this is not as simple as it sounds. Because it takes a finite amount of time to ask the Remote Concentrator to resume the local echoing for a particular line, it is possible that in the interval the Remote Concentrator has passed a character on to the CHIO without echoing it. This means that there is a possibility of one (or more) characters not being echoed. Hence, an algorithm is needed that can synchronize the resumption of the local echo.

Because of the Valparaiso Observation the output subroutine knows that if block number N is being input, it can send a message to the Remote Concentrator telling it to resume local echoing with block number N plus the Valparaiso round trip time. The period between the current input block (n) and the Resume Echo Block number (T = N + VRTT) is a period of hiatus. This period includes the first block (N), but not the last (T).

Any characters coming in after the hiatus will be assumed to have been locally echoed. Characters received before the block has been processed are assumed by both computers to have not been locally echoed.

Characters coming in during the hiatus are assumed to not be locally echoed by the Remote Concentrator. If a character comes in during hiatus, both processors act as if a Resume Echo Character had not been sent. This case is detected in the CHIO as follows (see below for the RC's viewpoint). Whenever a Resume Echo Character is sent to the Remote Concentrator a field in the CPU line table (LVB) is set to the Valparaiso round trip time. This is decremented for each input block received (as long as it is greater than 0.) If a character is input and LVB is not zero, then the CLE bit is reset.

There is one annoying case that has not been considered. If a character for a line is input at the beginning of block N, and a little later the CHIO decides to resume local echoing on that line, it cannot do so because the Remote Concentrator will ignore the REC (since it knows that it sent a character in block N).

The situation is detected by setting LVB to -1 whenever a character is taken out of HARRY. When the LVBs are decremented by 1 at the end of a block, each LVB is checked to see if it is -1, and if it is, LVB is zeroed.

Meanwhile back at the output multiplexor, if LVB is -1 and the line is trying to resume local echo a SNULL is sent. When the time comes for the next character to be sent, the situation should have improved.

Whenever REC is sent, it is followed by the block number being input (mod 128). Whenever this pair of characters is sent they are treated as one character for the specified device by the high speed line scanning algorithm.

The condition for sending REC to the RC is that a RSTB must have failed to return any characters to the CPU; no characters since then can have been input from the device, and input and output lines must both be empty. Detection of this is facilitated by setting a bit (RSTF) if an RSTB fails, and resetting it for any input. RSTF is checked if an RCH from the output line fails. It would be possible to slightly increase the useful output bandwidth by trying to encode the echo block number in the REC, but a simple calculation shows that a 5% improvement is about the best that could be expected (assuming a 500 user system).

Now lets look at things from the point of view of the RC. Whenever a device sends a character to the CHIO, LVB in the device table is set to VRTT. Whenever a new block is output to the CHIO, LVB for each teletype is decremented by one unless it is already zero. Let us consider the possibilities when REC is received on the input line. The REC has the Resume Echo Block number (REBN) following it. As you remember, this is the block which was being input when the REC was sent. The block currently being output is CBO. Observe that CBO + LVB - VRTT is the block number of the most recent block containing an input character.

It is constant as long as LVB $< \emptyset$. There are two possibilities:

1) CBO + LVB - VRTT $<$ REBN

2) CBO + LVB - VRTT $\geq$ REBN

Case 1 is the normal case. The last block sent to the CPU was received before it sent the REC. Local echoing may be resumed, and the Local Echo Bit is set. The Remote Concentrator may not send the CHIO input from this line at least until block REBN + VRTT is being sent to the CHIO. This is done by setting a bit in the line table called POL, and storing REBN + VRTT - CBO +1 in LVB. Each time a block is sent to the CHIO (as we stated previously) LVB is decremented by 1. When LVB is set to zero by this process, POL is reset (as was not previously stated). The input line scanner will ignore all lines that have the POL bit set. This postpones any input at least until block REBN + VRTT is being input to the EFCL. The first character input buffer in the remote concentrator absorbs the slop.

In Case 2 REBN + VRTT is less or equal to the block being output. That means that a character was sent to the CHIO more recently than REBN. This means that the REC has arrived during the period of hiatus, because a character was sent to the CHIO at (about) the same time as the CHIO sent the REC. The Remote Concentrator (like the CHIO) pretends the character was never received and does not attempt to resume local echo.

Outputting to A Device in Local Echo Mode

Outputting to a device that is in local echo mode is abnormal. Normally if the process expects input it will not generate any more output until a break character has been input. Local echo mode is turned off at this time. Thus, when the computer generates output, the system is no longer in local echo mode. The previous section specifies when local echo is normally resumed.

A process has to change its mind for it to output to a device that is in local echo mode. There are several ways that this could happen, but the most common, and one useful as an example, is for a timer to timeout and for the process to send a message to the device as a result. An obvious example is for the process to say:

TYPE IN

and if the user does not reply in ten seconds for it to type:

PLEASE TYPE IN

This is all well and good--but suppose that just when the computer typed the second message was when the user responded to the first one. For our example, assume that he responds with the alphabet. If he was on the 940 system the result would be as follows (assuming the computer's messages were followed and preceeded by carriage returns).

TYPE IN
A
PLEASE TYPE IN
BC

Things would be even worse if the carriage returns were not there.

At first glance it might seem that this is what we would like the M-1 system to do; but a little analysis is in order. Assuming that the program is a well-behaved program it knows that this problem exists, and it presumably has a plan for dealing with it.

The control character TAG is now introduced. It can be sent by a CPU program to a device. If this character (or any character) is output to a line, CLF, Local Echo Mode is turned off. When the character (or any other character) is received by the Remote Concentrator, Local Echo Mode is turned off, and the TAG is inserted into the input buffer for the line to be echoed back to the CPU program. There are several ways to use TAG to minimize problems generated by trying to type out while the user is typing in.

One possible method is now outlined. A process first reads any characters from the input buffer and outputs the TAG character. It then reads its input until it finds the TAG character (which is always treated as a break character). The process knows the characters that preceeded it were typed during the hiatus and can now decide to not output a message after all. It would then cause local echo mode to be resumed by characters from that time until it gets a fail return. Alternately, it could output the second message anyway and read the user's response to it.

If the process intends to accept any character input during the hiatus it might echo those characters that it received before the TAG, as well as letting those received after the TAG be echoed normally. This makes the output neater.

A problem of some subtlety presents itself. Just as the process outputs the TAG the user types in several characters, some appearing just before the TAG in the input string, and some just after the TAG. The process then types out a message which requests some response. The process must have a method of differentiating those characters that were typed in while the message was being output but not as a result of the message, from those that were typed during or after the message was typed and as a result of it. The solution is quite simple. If a process makes the assumption that it will take the user about eight character times to adjust to this new situation, a second TAG can be inserted just after the T in "PLEASE TYPE IN". All of the characters from the first TAG to the second TAG can then be treated as if they were in the first category; and those after the second TAG can be treated as if they are in the second category.

There are other methods of utilizing these TAG characters. The output of the message does not need to be postponed until the TAG character has been returned to the process if the process outputs a message that is independent of what the user types. If the characters typed by the user before

the message was printed are to be treated the same as the characters typed by him just after the message was printed, then only one TAG character need appear (after the T in the message). There are other methods of using the TAG character, some of such subtlety and sophistication as to be useless.