To: Distribution

From: Mike Grady

Date: 02/23/76

Subject: MCS Problems with Input Interruptions

There exist several problems in MCS related to the interruption of a user's input with output and the asynchronous nature of these events. Some of these problems had solutions in the old tty software, others have always plagued us. The initial installation of MCS did not attempt to solve these problems since they were both rare and difficult to solve. Now that MCS has reached a fairly stable state, we propose some new solutions.

There are three basic problem areas:

I) Terminals which use shift characters

Multics supports certain EBCDIC terminals which require shift codes to display upper and lower case characters. If a user has depressed the shift key and is sending upper case letters when the system interrupts with output, the fact that the system returns the keyboard to the user in downshift state is not recorded in the input stream. Thus when the input is read by the process all characters after the upshift are seen in upper case, while this is not what the user actually typed.

To solve this problem, we propose that the 355 take over responsibilty for processing shift characters. For devices which use shift characters, the 355 will maintain a bit indicating the current case of the terminal and will mark each upshifted character with its "100" bit on. Besides inspecting each input character for upshift and downshift, the 355 will reset the bit to its lowercase state whenever the keyboard is addressed (i.e., just after the completion of the output). Since the keyboard is always addressed with a downshift character, the 355 will always know the correct state of the terminal. The 6180 will ignore all upshift and downshift characters, using the "100" bit of the character to determine its case.

II) Interaction of type-ahead and output column position

Currently, the updating of the column position due to user input is not done when the input is actually typed, but rather when it is read by the user's process. This results in occasional differences in the actual carriage position of the terminal and where the system thinks it is. If a user anticipates a cuestion (i.e., some output which leaves the carriage away from the left margin) and answers it with type-ahead, the carriage will actually be out somewhere away from the left margin while the system thinks it has returned to the left margin as a result of the typed input. This can result in delay timing problems with the next piece of output.

The solution to this problem is fairly simple. The updating of the column position must be done at the time the input is typed, and not when it is read. However, it must remain in the 6180 since that is where the column position information is kept. Input is sent to the 6180 just after the newline is typed (with an indicator that it contains a break character), so the the 6180 interrupt handler can reset the column position. An assumption is made that all such input interrupts cause the column position to return to zero, and no test is made of the actual input.

III) Partial input lines interrupted by output

This is both the most bothersome problem and the most difficult to solve. When conversion of output begins, the system assumes it knows the position of the carriage (e.g., at the left margin). If the user has typed a partial input line, the carriage will not be where the system thought and two problems can occur. First, the length of the first line of output plus the partial input line may be greater the the line length of the terminal. This would cause information typed after the end of the line to be garbled or lost. Second, the first newline in the output may not have enough delay timing characters after it since the carriage was farther to the right than the system thought.

The best possible solution would be to have tty_write interrogate the 355 to obtain the current column position before starting the conversion of each piece of output. This is a non-trivial operation since the 355 cannot report the column position immediately, but must stop the channel, scoop up the input, scan it and report the position. This would require tty_write to wait for the response (loop?) and would add two extra interrupts to each output preparation attempt by tty_write. This is felt to be far to expensive to justify the few cases it would benefit and an alternate solution is proposed here.

The output will be converted by tty_write with the assumption that the column position is where it was left by

either tty_write or the last complete piece of input. When the 355 receives some new output for a channel, it will check the current column position and if it is non-zero the 355 will send a newline plus delays to the terminal followed by the output. The output will start in the left margin as tty_write expected, and the correct number of delays will be inserted for the first newline.

Certain Multics functions (e.g., send_message) already insert newline characters at the beginning of their output to avoid these problems. These commands should be converted not to do so, since the 355 will be doing it automatically.

Two new features will be added with this change to assist recovery from these interruptions. A new mode, replay, will cause the accumulated input to be written back to the terminal at the end of the output, so the user may see how much of his input was actually received by the system and where he must restart his typing.

A second mode, polite, will disable new output from being written on the terminal if there is a partial input line. As soon as the input line is completed, the output will be sent. If the input line is not completed within 30 seconds after the time the output arrives at the 355, it will be sent to the terminal anyway and the replay option will come into effect.