ECE3710 Makeup Project

One make-up project is allowed per student during the semester. The make-up project is worth ½ of the points missed on one midterm exam. For example, if your exam score (after scaling) is 72%, then you may earn an additional (100-72) ÷ 2 = 14 exam points by successfully completing the project and demonstrating it to me.

The project this semester is a 4-function calculator using the RS-232 serial port on the C8052F020 microcontroller board. The requirements are as follows:

1. The calculator shall communicate using RS232 protocol, 9600 baud, 1 stop bit, no parity.
2. The calculator shall recognize and echo the numbers '0'-'9', the symbols '+', '-', '*', '/' and '='. All other characters shall be ignored.
3. As soon as the '=' key is pressed, the input is checked to verify that it is of the form <number><operator><number>'=' (where <number> consists of one or more digits and <operator> is '+', '-', '*' or '/').
4. If the input does not have the right form, or if a division by zero is requested, the calculator responds with a error string (e.g. "ERROR") followed by a carriage return ($0D_{16}$) and a line feed ($0A_{16}$).
5. If the input has the right form and division by zero is not requested, the calculator shall calculate the result using 32-bit unsigned integer arithmetic. If an overflow occurs, or if the result should be negative, the calculator responds with an overflow string (e.g. "OVERFLOW") followed by a carriage return ($0D_{16}$) and a line feed ($0A_{16}$).
6. If the result of the calculation does not overflow and is non-negative, the calculator outputs the result (in base 10 without leading zeros) using the digits '0'-'9' followed by a carriage return and a line feed.
7. After the carriage return and line feed are sent, the calculator shall be prepared to accept another line of input.
8. The software shall be written in assembly language.

Note that you may use different error strings than those provided. You may also use different error strings for syntax error, division by zero, overflow and underflow.

Overachievers may also support the mod operator ("%"), which returns the remainder of a division instead of the quotient. Overachievers may also make their calculator support negative numbers. In which case, no error would be generated for negative numbers and the valid number range would be $-2^{-31}..2^{31}-1$ instead of $0..2^{32}-1$. You don't get more points, but it makes the project more fun.