Exercise 3 Documentation

## Summary of Task:

Exercise 3 should take a string from the terminal emulator which is connected to the SCI1 input port, then be read in through the SCI to be stored in memory. This string stored in memory should then be outputted through the SCI port and displayed on the terminal. The carriage break character should be used to denote the end of the string. The string should print out once per second.

## Program Summary:

* The program reserves 300 bytes of memory at address inpstr
* The program sets the baud rate at 9600 by writing to the SCI1BDH/DL registers
* Loads the address of inpstr to register x
* Using a subroutine the program jumps to the receive subroutine
  + The control register 2 is set to receiving enabled
  + When the mask for RDRF empty is 1, the subroutine will run, not moving to the next character until this mask is met.
  + The data in the SCI1DRL register is stored in reg b
  + This is compared to the carriage character value
  + If 0, return from the function and add the carriage character to the current memory address of inpstr
  + Else store the value of the char in reg b at current address position in x, increment x
* Using a subroutine there is a delay of 1 second.
* Using a subroutine the program jumps to the transmit subroutine.
  + The control register 2 is set to transmitting enabled.
  + When the mask for RTDE empty is 1, the subroutine will run, not moving to the next character until this mask is met.
  + The data in the address of x in moved to SCI1DRL register.
  + This is compared to the carriage character value.
  + If 0, return from the subroutine as this is the end of the string, only return when transmission is complete.
  + Else store loop through the reserved memory until the carriage character.

## Discussion Questions:

* What problems can you see arising from the use of polling when dealing with data input?
  + The main problem will polling is it limits the device to waiting for the data to be ready before it can move on. This is fine if time is not important and this is the only task we want to achieve. However if we want to perform other tasks at the same time, this is not possible as polling relies on a sequential order of events. Another Main issues is the problem arising when timing is critical, as polling does take a finite amount of time, this can have negative effects on the program.
* What happens if there are more characters input than there is space to store them?
  + These characters will flow onto the next memory addresses. This is ok if there is nothing else being stored in memory, but in our example we have another space for memory where we store the numbers used for the seven 7. If there isn’t enough space defined, the string will fill this space, then this will be overwritten by characters or the characters will be later sent to the seven seg. Further more when this str goes to be sent out via serial, it could be overwritten by numbers.

## Testing:

* Are the SCI registers for baud rate, control and status being set correctly?
  + Check that SCI1BDH is set to #00
  + Check that SCI1BDL is set #156 (baud rate)
  + Check that SCI1CR2 has a binary value of #%00001100
  + Check status register for errors when unexpected behavior occurs.
* Printing to the terminal
  + Check that the sub-routine is being called and run by setting a breakpoint in the TE subroutine.
  + Check that register X is loading the start of the strings location
  + Check that the branch until the TDRF flag is being set and the code runs past this branch by setting a breakpoint after this conditional branch.
  + Use Putty to verify serial signals can be sent by sending chars from string.
    - If this doesn’t work, verify comm ports.
    - Verify serial registers are all for port 1.
    - Check code loaded successfully onto the board.
  + Check that the terminating character is causing the code to break from the sub-routine by setting a breakpoint before the RTS.
  + If there are errors use the status registers to check for framing, parity, or other others.
* Inputting a string from the terminal
  + Check that the sub-routine is being called and run by setting a breakpoint in the RE subroutine.
  + Check that register X is loading the start of the memory’s location
  + Check that the branch until the RDRF flag is being set and the code runs past this branch by setting a breakpoint after this conditional branch.
  + Check that the SCI1DRL is loading a value into a register.
  + Check that ascii values are entering as expected by using ascii table.
  + Make sure X is incrementing as desired by using a breakpoint to the step through the increment step.
  + Check that the terminating character is causing the code to break from the sub-routine by setting a breakpoint before the RTS.
  + If there are errors use the status registers to check for framing, parity, or other others.

## Diagram:

