;---------------- Lab 10 Multi-Byte Key based symmetric XOR encryption ---------------

;variable and constant definitions

keyBytesRAMaddress EQU 0x40 ;symbolic constant for base address of encryption key in RAM

keyLength EQU 0x30 ;variable to track length of key

keyvalIndex EQU 0xe0 ;variable to index the keyval constant array

;keyvalIndex is also an alias for accumulator

;begin section from lab 9

;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv

jmp main ;jump past interrupt vector table

org 0x0030 ;put main program at rom location 0x0030

main:

;---------------- Initialization/configuration ----------------;

;keyval variable no longer used

;mov keyval, #0x23 ;load the keyval variable with encryption key

mov tmod, #0x20 ;config timer 1 mode 2

mov scon, #0x50 ;config serial 8-data, 1 start, 1 stop, no parity

mov th1, #0xFD ;9600 baud

setb tr1 ;start timer 1 to enable serial communication

;end section from lab 9

;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

;In the following section load the key bytes from ROM into RAM

mov r0, #keyBytesRAMaddress ;initialize RAM pointer

mov dptr, #keyvals2 ;initialize ROM pointer

mov keyvalIndex, #0x00 ;initialize keyvalIndex

LoadKey:

push keyvalIndex ;preserve keyvalIndex variable

movc a, @a+dptr ;load byte of key from ROM

cjne a, #0x00, notNull ;check for null terminating character

jmp LoadDone ;if null is found, enter main\_loop

notNull:

mov @r0, a ;put byte of key into ram

pop keyvalIndex ;restore keyvalIndex variable

inc keyvalIndex ;increment keyvalIndex

inc r0 ;increment RAM pointer

jmp LoadKey ;continue the loop

LoadDone:

mov @r0, #0x00 ;append null char to string

mov r0, #keyBytesRAMaddress ;re-initialize RAM pointer

;---------------- END of Initialization/configuration ----------------;

;begin section from lab 9

;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv

mainloop:

jnb ri, $ ;wait to receive a char

call getchar ;char received, get it!

;cjne a, #0x00, encrypt ;check for null character in string

cjne a, #0x00, checkKeyVal ;check for null character in string

jmp terminate ;terminate program if null character is received

;end section from lab 9

;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

checkKeyVal:

cjne @r0, #0x00, Encrypt ;go to Encrypt if keyVal is not null

mov r0, #keyBytesRAMaddress ;re-initialize RAM pointer

;begin section from lab 9

;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv

Encrypt:

xrl a, @r0 ;encrypt the character contained in the accumulator

call writechar ;write the encrypted character

inc r0

jmp mainloop

terminate:

mov a, #0x00 ;load null character into accumulator

call writechar ;append the null character to text output

sjmp $ ;halt

;----------- getchar ----------;

;subroutine receives nothing before it is called

;writes the character to the serial console

;returns a byte in the accumulator

getchar:

mov a, sbuf ;get serial data (char)

clr ri ;acknowledge data received

ret ;return from subroutine call

;----------- writechar ----------;

;receives byte or character

;reads a character that has been received serially

;returns the c

writechar:

mov sbuf, a ;send data (char) serially

jnb ti, $ ;wait until data is sent

clr ti ;acknowledge data has been sent

ret ;return from subroutine call

;end section from lab 9

;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

;multibyte keys are defined below, only one will be used at a time

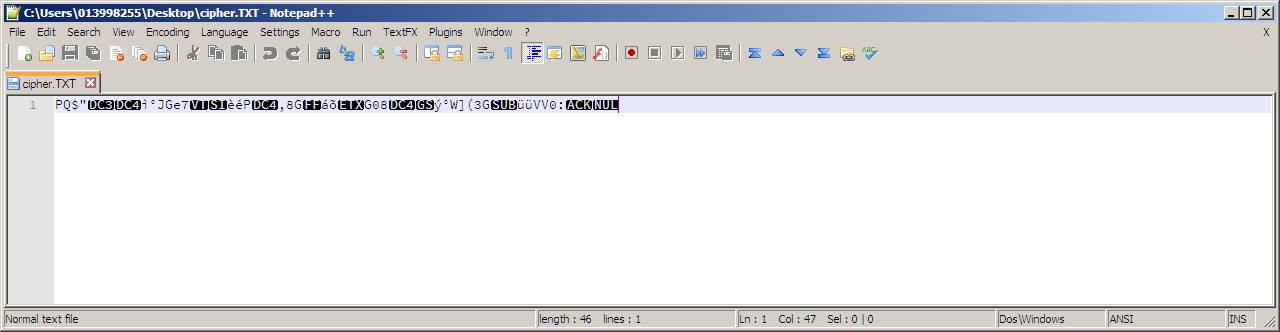
org 0x200

keyvals: db '12345678',0

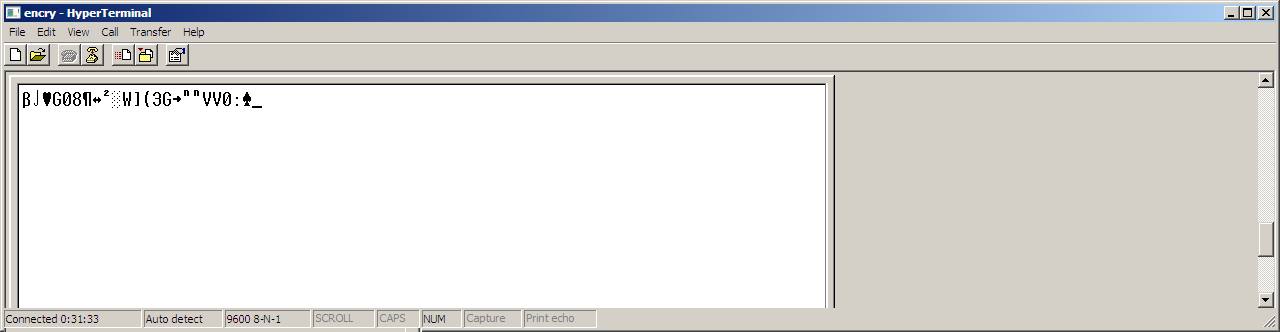
keyvals2: db 0x23, 0x34, 0x45, 0x56, 0x67, 0x78,0x89,0x90, 0x00

end

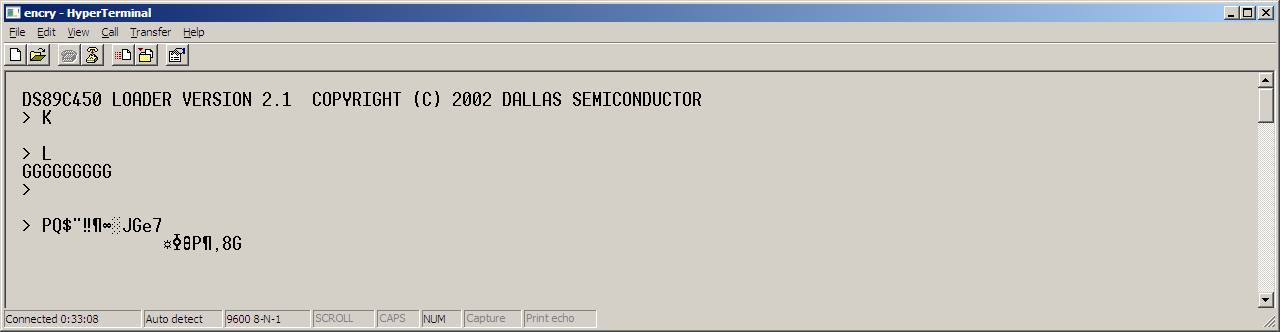
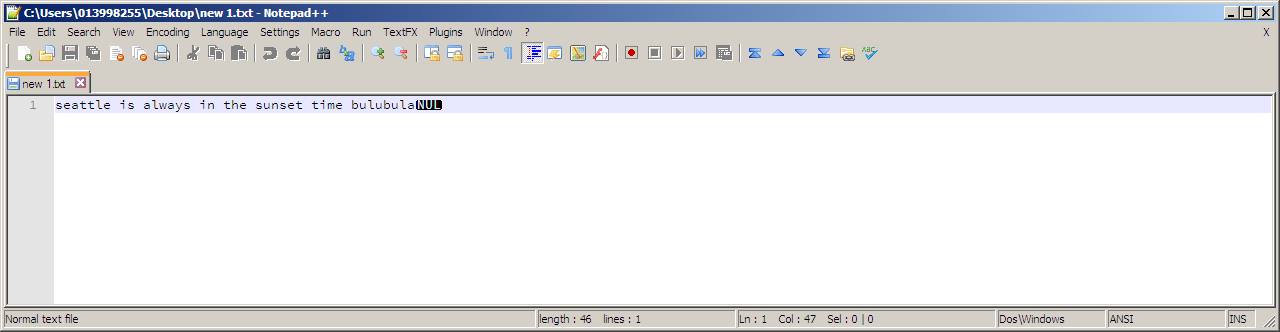
cipher text:



Cipher text in Hyper terminal:



Plain text:



Manual encryption:

