QUESTION 1

```
;/----------------------PROGRAM DESCRIPTION----------------------\;

;RECEIVE A CHARACTER FROM PC AND TURN ON AN LED TO INDICATE RECEPTION

   ;PRESS A BUTTON TO ENCRYPT RECEIVED CHARACTER AND SEND IT BACK

    ;9600 BPS, 8 DATA BITS, 1 START BIT, 1 STOP BIT, NO PARITY

              ;USES RX INTERRUPT AND INT0 INTERRUPT

;/------------------------------LIST------------------------------\;

LIST P=18F2420, MM=OFF, R=HEX, ST=OFF, X=OFF

;/--------------------------CONFIG BITS---------------------------\;

; CONFIG1H

  CONFIG  OSC = HS              ; Oscillator Selection bits (HS oscillator)

  CONFIG  FCMEN = OFF           ; Fail-Safe Clock Monitor Enable bit (Fail-Safe Clock Monitor
disabled)

  CONFIG  IESO = OFF            ; Internal/External Oscillator Switchover bit (Oscillator
Switchover mode disabled)

; CONFIG2L

  CONFIG  PWRT = OFF            ; Power-up Timer Enable bit (PWRT disabled)

  CONFIG  BOREN = OFF           ; Brown-out Reset Enable bits (Brown-out Reset disabled in hardware
and software)

  CONFIG  BORV = 3              ; Brown Out Reset Voltage bits (Minimum setting)

; CONFIG2H

  CONFIG  WDT = OFF             ; Watchdog Timer Enable bit (WDT disabled (control is placed on the
SWDTEN bit))

  CONFIG  WDTPS = 32768         ; Watchdog Timer Postscale Select bits (1:32768)

; CONFIG3H

  CONFIG  CCP2MX = PORTC        ; CCP2 MUX bit (CCP2 input/output is multiplexed with RC1)

  CONFIG  PBADEN = OFF          ; PORTB A/D Enable bit (PORTB<4:0> pins are configured as digital
I/O on Reset)

  CONFIG  LPT1OSC = OFF         ; Low-Power Timer1 Oscillator Enable bit (Timer1 configured for
higher power operation)

  CONFIG  MCLRE = ON            ; MCLR Pin Enable bit (MCLR pin enabled; RE3 input pin disabled)

; CONFIG4L

  CONFIG  STVREN = OFF          ; Stack Full/Underflow Reset Enable bit (Stack full/underflow will
not cause Reset)

  CONFIG  LVP = OFF             ; Single-Supply ICSP Enable bit (Single-Supply ICSP disabled)

  CONFIG  XINST = OFF           ; Extended Instruction Set Enable bit (Instruction set extension
and Indexed Addressing mode disabled (Legacy mode))

; CONFIG5L

  CONFIG  CP0 = OFF             ; Code Protection bit (Block 0 (000800-001FFFh) not code-protected)

  CONFIG  CP1 = OFF             ; Code Protection bit (Block 1 (002000-003FFFh) not code-protected)
```

```
; CONFIG5H

  CONFIG  CPB = OFF             ; Boot Block Code Protection bit (Boot block (000000-0007FFh) not
code-protected)

  CONFIG  CPD = OFF             ; Data EEPROM Code Protection bit (Data EEPROM not code-protected)

; CONFIG6L

  CONFIG  WRT0 = OFF            ; Write Protection bit (Block 0 (000800-001FFFh) not write-
protected)

  CONFIG  WRT1 = OFF            ; Write Protection bit (Block 1 (002000-003FFFh) not write-
protected)

; CONFIG6H

  CONFIG  WRTC = OFF            ; Configuration Register Write Protection bit (Configuration
registers (300000-3000FFh) not write-protected)

  CONFIG  WRTB = OFF            ; Boot Block Write Protection bit (Boot block (000000-0007FFh) not
write-protected)

  CONFIG  WRTD = OFF            ; Data EEPROM Write Protection bit (Data EEPROM not write-
protected)

; CONFIG7L

  CONFIG  EBTR0 = OFF           ; Table Read Protection bit (Block 0 (000800-001FFFh) not protected
from table reads executed in other blocks)

  CONFIG  EBTR1 = OFF           ; Table Read Protection bit (Block 1 (002000-003FFFh) not protected
from table reads executed in other blocks)

; CONFIG7H

  CONFIG  EBTRB = OFF           ; Boot Block Table Read Protection bit (Boot block (000000-0007FFh)
not protected from table reads executed in other blocks)

;/-----------------INCLUDE LIBRARY FOR PIC18F2420----------------\;

#include <p18f2420.inc>

VAR    EQU 0x0A                  ;RAM location to copy received data to. VAR for 'variable'

;/---------------SETUP FOR POWER UP AND INTERRUPTS---------------\;

       ORG 0x00

       GOTO START                ;Go to beginning of program


       ORG 0x08

       BTFSC PIR1, RCIF          ;Execute RX_ISR if received data; RCIF = 1

       BRA RX_ISR

       BTFSS INTCON, INT0IF      ;Execute CIPHER_ISR if pushbutton was pressed; INT0IF = 1

       RETFIE

       BRA CIPHER_ISR


       ORG 0x18

       RETFIE
```

```
;/--------------------START OF ACTUAL PROGRAM--------------------\;
```

START

```
;/SETUP
        ;/Initialize RC0 to 0V
        BCF PORTC, 0              ;LED should be off since no data has been received yet

        ;/Configure TXSTA register
        MOVLW B'00100000'         ;Enable transmit, 8-bit transmission
        MOVWF TXSTA               ;Asynchronous mode

        ;/Configure RCSTA register
        MOVLW B'10010000'         ;Enable serial port, continuously receive 8 bit data, no framing
                                   error bit, no overrun error bit
        MOVWF RCSTA

        ;/Configure baud rate settings
        MOVLW D'25'               ;9600 bps. [(16 MHz / 64) / 9600] - 1 = 25.04 -> 25
        MOVWF SPBRG

        ;/Clear WREG
        CLRF WREG                 ;WREG is used to test if button has been pressed if no data has
                                   been received
        ;/Make TX pin an output pin
        BCF TRISC, TX

        ;/Make RX pin an input pin
        BSF TRISC, RX

        ;/Make RC0 pin an output pin for LED receive indicator
        BCF TRISC, 0

        ;/Make INT0 an input pin to sense push button to send data
        BSF TRISB, INT0

        ;/Enable interrupts
        BSF PIE1, RCIE            ;RX interrupt
        BSF INTCON, INT0IE        ;INT0 interrupt
        BSF INTCON, PEIE          ;Peripheral interrupt for COM port
        BSF INTCON, GIE           ;Officially allow interrupts to occur
```

```
        ;/Wait for an interrupt flag to be raised
        BRA $
;/------------------INTERRUPT SERVICE ROUTINES--------------------\;
RX_ISR                  ;RX_ISR for "receive interrupt service routine"
        MOVFF RCREG, VAR    ;Copy received character to RAM
        BSF PORTC, 0        ;Light up LED to indicate data has been received
        RCALL MSG1          ;Print "Plaintext: " message with received character
114     RETFIE              ;Exit RX_ISR. Return to BRA $. Reenable global interrupts

CIPHER_ISR              ;CIPHER_ISR for "cipher interrupt service routine"
        MOVF WREG           ;Exit CIPHER_ISR if button is pressed if no data has been received
        BZ EXIT             ;WREG != 0 if data has been received. Thus, exit CIPHER_ISR if Z flag = 1
        RCALL MSG2          ;Print "Ciphertext: " message with ciphered character
119     BCF PORTC, 0        ;Turn off LED
        CLRF WREG           ;Clear WREG. Only send character once
EXIT    BCF INTCON, INT0IF  ;Reset INT0 interrupt flag
        RETFIE              ;Exit CIPHER_ISR. Return to BRA $. Reenable global interrupts
;/-------------------------SUBROUTINES--------------------------\;
MSG1
;/Print "Plaintext: " message
        ;/Load Table Pointer with address of PLAINTEXT
        MOVLW upper(PLAINTEXT)
        MOVWF TBLPTRU
        MOVLW high(PLAINTEXT)
        MOVWF TBLPTRH
        MOVLW low(PLAINTEXT)
        MOVWF TBLPTRL

;/Send "Plaintext: "
        ;/Read from table, increment pointer, then send character
READ1   TBLRD*+
        MOVF TABLAT, WREG   ;WREG = TABLAT
        BZ SEND_PLAIN       ;Go to SEND_PLAIN until WREG = NULL
        RCALL SEND          ;Send character
        BRA READ1           ;Repeat until null
```

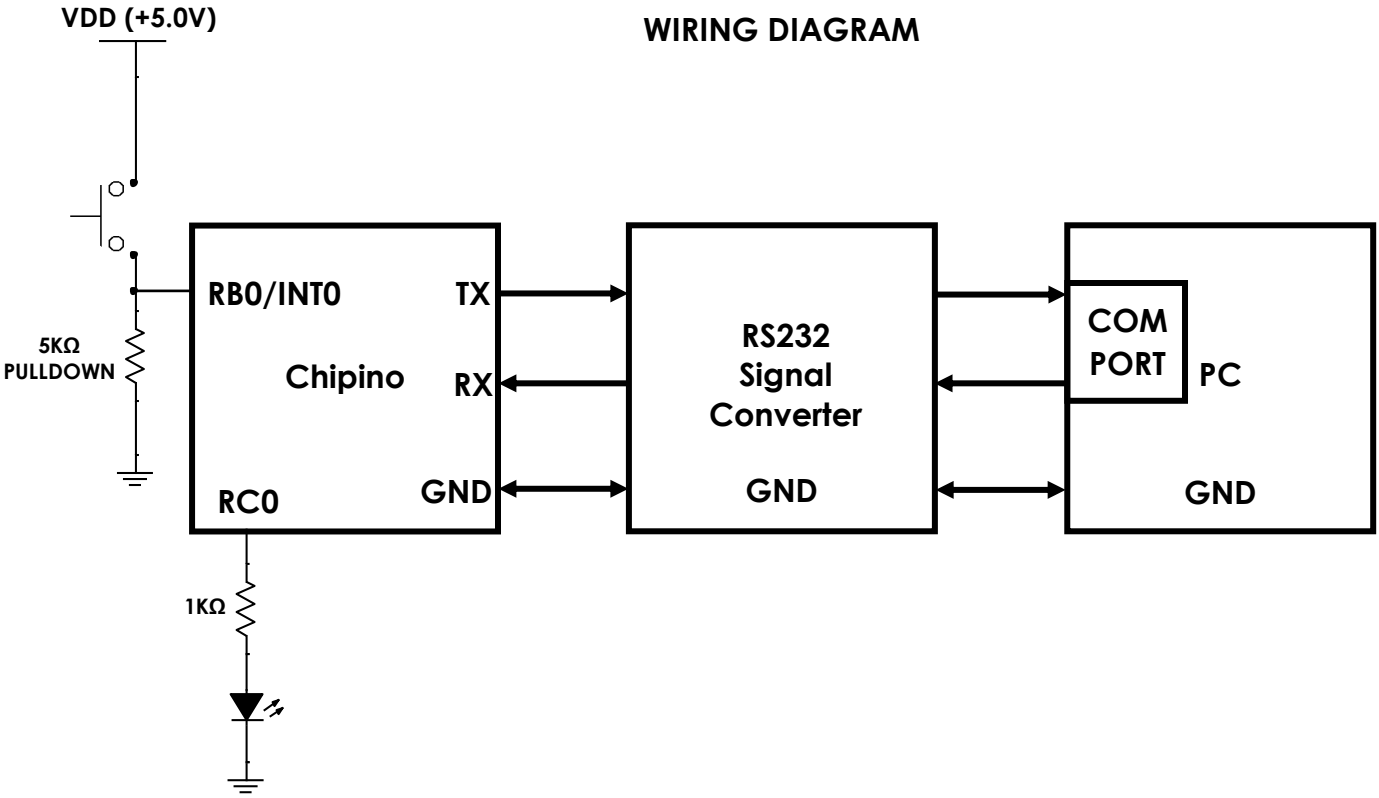
```
;/Send received character
SEND_PLAIN    MOVF VAR, WREG      ;Copy received character to WREG
              RCALL SEND          ;Transmit received character
              RCALL LFCR          ;Transmit Newline and Carriage Return
              RETURN              ;Exit MSG1 subroutine. Return to line 114


;/-------------------------------------------------------------\;
```

MSG2

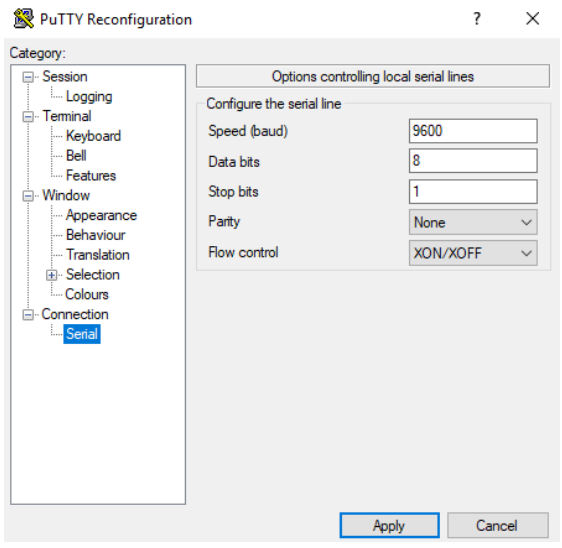
```
;/Print "Ciphertext: " message
       ;/Load Table Pointer with address of CIPHERTEXT
       MOVLW upper(CIPHERTEXT)
       MOVWF TBLPTRU
       MOVLW high(CIPHERTEXT)
       MOVWF TBLPTRH
       MOVLW low(CIPHERTEXT)
       MOVWF TBLPTRL

;/Send "Ciphertext: "
       ;/Read from table, increment pointer, then send character
READ2  TBLRD*+
       MOVF TABLAT, WREG   ;WREG = TABLAT
       BZ SEND_CIPHER      ;Go to SEND_CIPHER until WREG = NULL
       RCALL SEND          ;Send character
       BRA READ2           ;Repeat until null

;/Send Ciphered character
SEND_CIPHER   MOVF VAR, WREG      ;Copy received character to WREG
              XORLW B'01100111'   ;Arbitrary cipher algorithm
              RCALL SEND          ;Send ciphered character
              RCALL LFCR          ;Newline and Carriage Return
              RETURN              ;Exit MSG2 subroutine. Return to line 119
```

```
;/-----------------------------------------------------------------\;
LFCR                        ;LFCR for "Line Feed/Carriage Return"
;/Transmit Newline and Carriage Return
       MOVLW H'D'           ;Transmit Newline/Line Feed
       RCALL SEND
       MOVLW H'A'           ;Transmit Carriage Return
       RCALL SEND
       RETURN
;/-----------------------------------------------------------------\;
SEND
;/Transmit subroutine
L1     BTFSS PIR1, TXIF     ;Make sure the last bit of the previous frame has been sent
       BRA L1
       MOVWF TXREG          ;Transmit character
       RETURN
;/-------------------------DEFINITIONS-------------------------\;
PLAINTEXT  DB "Plaintext: ", 0
CIPHERTEXT DB "Ciphertext: ", 0
END
```

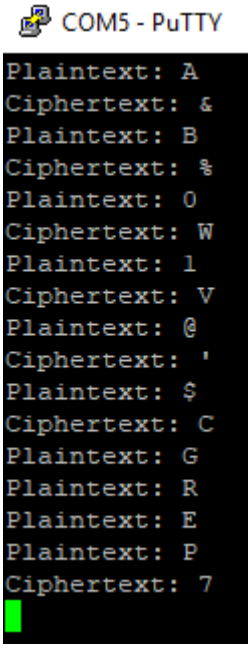
WIRING DIAGRAM



Screenshots from PuTTY terminal emulator



Serial COM settings



Sample Output