```
// PIC18F4520 Configuration Bit Settings

// 'C' source line config statements

// CONFIG1H
#pragma config OSC = INTIO67    // Oscillator Selection bits (Internal oscillator block, port
function on RA6 and RA7)
#pragma config FCMEN = OFF      // Fail-Safe Clock Monitor Enable bit (Fail-Safe Clock Monitor
disabled)
#pragma config IESO = OFF       // Internal/External Oscillator Switchover bit (Oscillator
Switchover mode disabled)

// CONFIG2L
#pragma config PWRT = OFF       // Power-up Timer Enable bit (PWRT disabled)
#pragma config BOREN = SBORDIS  // Brown-out Reset Enable bits (Brown-out Reset enabled
in hardware only (SBOREN is disabled))
#pragma config BORV = 3         // Brown Out Reset Voltage bits (Minimum setting)

// CONFIG2H
#pragma config WDT = OFF        // Watchdog Timer Enable bit (WDT disabled (control is placed
on the SWDTEN bit))
#pragma config WDTPS = 32768    // Watchdog Timer Postscale Select bits (1:32768)

// CONFIG3H
#pragma config CCP2MX = PORTC   // CCP2 MUX bit (CCP2 input/output is multiplexed with
RC1)
#pragma config PBADEN = OFF     // PORTB A/D Enable bit (PORTB<4:0> pins are configured as
digital I/O on Reset)
#pragma config LPT1OSC = OFF    // Low-Power Timer1 Oscillator Enable bit (Timer1 configured
for higher power operation)
#pragma config MCLRE = ON      // MCLR Pin Enable bit (RE3 input pin enabled; MCLR disabled)

// CONFIG4L
#pragma config STVREN = ON      // Stack Full/Underflow Reset Enable bit (Stack full/underflow
will cause Reset)
#pragma config LVP = OFF        // Single-Supply ICSP Enable bit (Single-Supply ICSP disabled)
#pragma config XINST = OFF      // Extended Instruction Set Enable bit (Instruction set extension
and Indexed Addressing mode disabled (Legacy mode))

// CONFIG5L
#pragma config CP0 = OFF        // Code Protection bit (Block 0 (000800-001FFFh) not
code-protected)
#pragma config CP1 = OFF        // Code Protection bit (Block 1 (002000-003FFFh) not
code-protected)
#pragma config CP2 = OFF        // Code Protection bit (Block 2 (004000-005FFFh) not
code-protected)
#pragma config CP3 = OFF        // Code Protection bit (Block 3 (006000-007FFFh) not
code-protected)

// CONFIG5H
#pragma config CPB = OFF        // Boot Block Code Protection bit (Boot block (000000-0007FFh)
not code-protected)
#pragma config CPD = OFF        // Data EEPROM Code Protection bit (Data EEPROM not
code-protected)

// CONFIG6L
#pragma config WRT0 = OFF       // Write Protection bit (Block 0 (000800-001FFFh) not
write-protected)
#pragma config WRT1 = OFF       // Write Protection bit (Block 1 (002000-003FFFh) not
write-protected)
#pragma config WRT2 = OFF       // Write Protection bit (Block 2 (004000-005FFFh) not
write-protected)
#pragma config WRT3 = OFF       // Write Protection bit (Block 3 (006000-007FFFh) not
write-protected)

// CONFIG6H
#pragma config WRTC = OFF       // Configuration Register Write Protection bit (Configuration
registers (300000-3000FFh) not write-protected)
#pragma config WRTB = OFF       // Boot Block Write Protection bit (Boot block
(000000-0007FFh) not write-protected)
#pragma config WRTD = OFF       // Data EEPROM Write Protection bit (Data EEPROM not
write-protected)

// CONFIG7L
#pragma config EBTR0 = OFF      // Table Read Protection bit (Block 0 (000800-001FFFh) not
protected from table reads executed in other blocks)
#pragma config EBTR1 = OFF      // Table Read Protection bit (Block 1 (002000-003FFFh) not
protected from table reads executed in other blocks)
#pragma config EBTR2 = OFF      // Table Read Protection bit (Block 2 (004000-005FFFh) not
protected from table reads executed in other blocks)
#pragma config EBTR3 = OFF      // Table Read Protection bit (Block 3 (006000-007FFFh) not
protected from table reads executed in other blocks)

// CONFIG7H
#pragma config EBTRB = OFF      // Boot Block Table Read Protection bit (Boot block
(000000-0007FFh) not protected from table reads executed in other blocks)

// #pragma config statements should precede project file includes.
// Use project enums instead of #define for ON and OFF.

#include <xc.h>

#define _XTAL_FREQ 4000000  // Define system clock frequency

void kaka (unsigned char);
void main(){
   OSCCON = 0xEF;
   TRISC6 = 0;

   TXSTA = 0x24;
   SPBRG = 25;
   RCSTA = 0x90;
```

```
    while(1){
        kaka('H');
        kaka('I');
        kaka('I');

        kaka('\r');
        kaka('\n');
        __delay_ms(300);
    }
}

void kaka(unsigned char c){
    while(PIR1bits.TXIF == 0);
    TXREG = c;
}
```