Practice 7

SERIAL TRANSIMISION (CHAR 'R')

Profesor: Paz Rodríguez Héctor Manuel

Grupo: 3CV3

Ricardo Ruiz Maldonado

ESCUELA SUPERIOR DE CÓMPUTO | INSTITUTO POLITÉCNICO NACIONAL

Theoretical reference

1 Introduction

USART:

Is a piece of computer hardware that translates data between parallel and serial forms. UARTs are commonly used in conjunction with communication standards such as EIA, RS-232, RS-422 or RS-485. The universal designation indicates that the data format and transmission speeds are configurable. The electric signaling levels and methods (such as differential signaling etc.) are handled by a driver circuit external to the UART.

A UART is usually an individual (or part of an) integrated circuit used for serial communications over a computer or peripheral device serial port. UARTs are now commonly included in microcontrollers. A dual UART, or DUART, combines two UARTs into a single chip. An octal UART or OCTART combines eight UARTs into one package, an example being the NXP SCC2698. Many modern ICs now come with a UART that can also communicate synchronously; these devices are called USARTs (universal synchronous/asynchronous receiver/transmitter).

USART Initialization:

The USART has to be initialized before any communication can take place. The initialization process normally consists of setting the baud rate, setting frame format and enabling the Transmitter or the Receiver depending on the usage. For interrupt driven USART operation, the Global Interrupt Flag should be cleared (and interrupts globally disabled) when doing the initialization.

Before doing a re-initialization with a changed baud rate or frame format, be sure that there are no ongoing transmissions during the period the registers are changed. The TXC Flag can be used to check that the Transmitter has completed all transfers and the RXC Flag can be used to check that there are no unread data in the receive buffer. Note that the TXC Flag must be cleared before each transmission (before UDR is written) if it is used for this purpose.

Data Reception – The USART Receiver:

The USART Receiver is enabled by writing the Receive Enable (RXEN) bit in the UCSRB Register to one. When the Receiver is enabled, the normal pin operation of the RxD pin is overridden by the USART and given the function as the Receiver's serial input. The baud rate, mode of operation and frame format must be set up once before any serial reception can be done. If synchronous operation is used, the clock on the XCK pin will be used as a transfer clock.

Receive Compete Flag and Interrupt:

The USART Receiver has one flag that indicates the receiver state.

The Receive Complete (RXC) Flag indicates if there are unread data present in the receive buffer. This flag is one when unread data exist in the receive buffer and zero when the receive buffer is empty (i.e., does not contain any unread data). If the Receiver is disabled (RXEN = 0), the receive buffer will be flushed and consequently the RXC bit will become zero.

When the Receive Complete Interrupt Enable (RXCIE) in UCSRB is set, the USART Receive Complete Interrupt will be executed as long as the RXC Flag is set (provided that global interrupts are enabled). When interrupt-driven data reception is used, the receive complete routine must read the received data from UDR in order to clear the RXC Flag, otherwise a new interrupt will occur once the interrupt routine terminates.

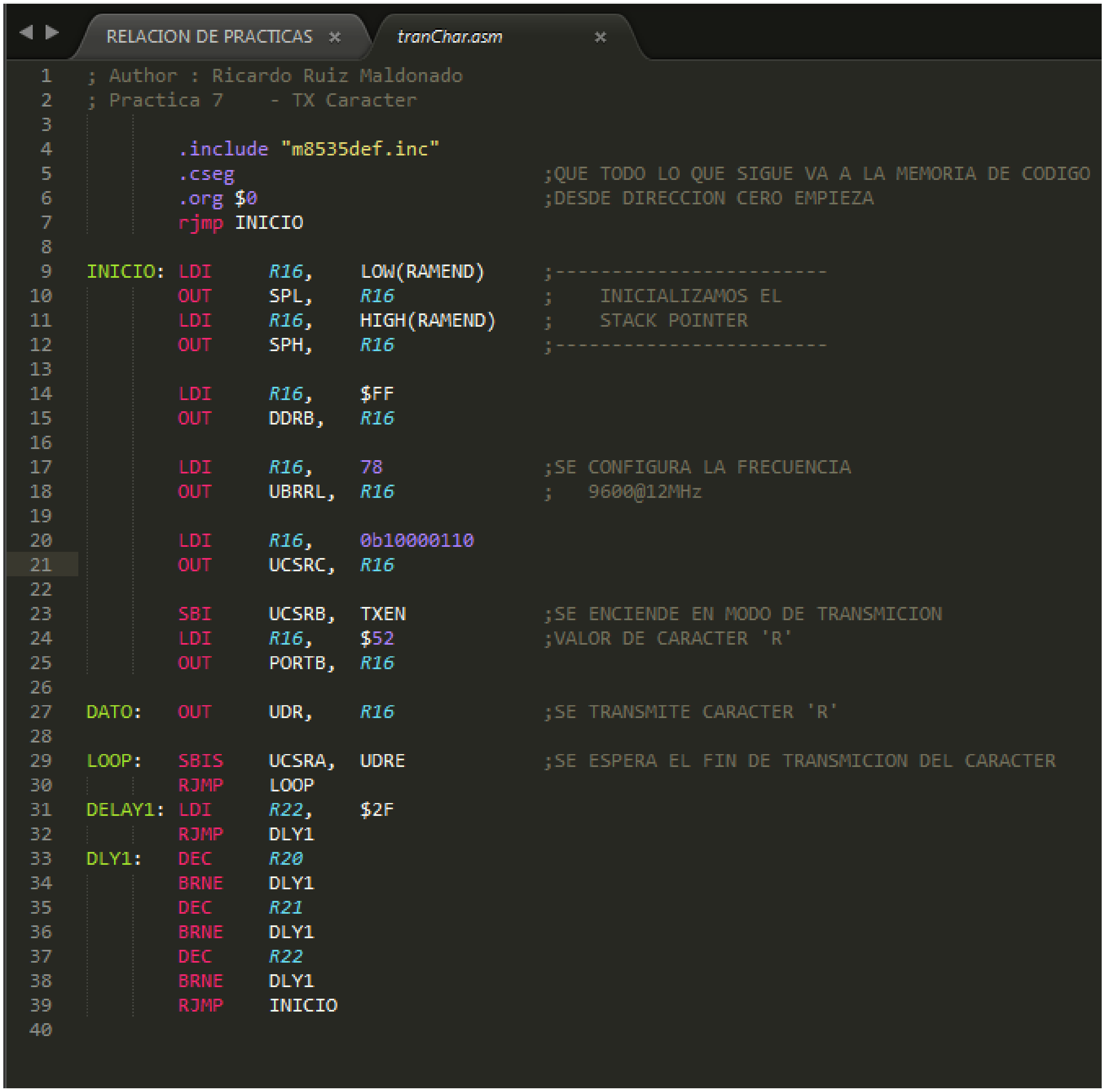
2 MATERIAL

- Pazuino

3 DEVELOPMENT

This example shows how to use the EUSART module microcontroller. The connection between the microcontroller and a PC is established in accordance with the standard RS-232 communication. The program works as follows. Each byte received via the serial communication is displayed when using the LEDs connected to PORTB and then automatically returns to the transmitter. The easiest way is to check the operation of the device in practice to use a standard Windows program called Hyper Terminal.

4 CODE



```
; Author : Ricardo Ruiz Maldonado
; Practica 7    - TX Caracter

        .include "m8535def.inc"
        .cseg                               ;QUE TODO LO QUE SIGUE VA A LA MEMORIA DE CODIGO
        .org $0                             ;DESDE DIRECCION CERO EMPIEZA
        rjmp INICIO

INICIO: LDI     R16,    LOW(RAMEND)         ;------------------------
        OUT     SPL,    R16                 ;    INICIALIZAMOS EL
        LDI     R16,    HIGH(RAMEND)        ;    STACK POINTER
        OUT     SPH,    R16                 ;------------------------

        LDI     R16,    $FF
        OUT     DDRB,   R16

        LDI     R16,    78                  ;SE CONFIGURA LA FRECUENCIA
        OUT     UBRRL,  R16                 ;   9600@12MHz

        LDI     R16,    0b10000110
        OUT     UCSRC,  R16

        SBI     UCSRB,  TXEN                ;SE ENCIENDE EN MODO DE TRANSMICION
        LDI     R16,    $52                 ;VALOR DE CARACTER 'R'
        OUT     PORTB,  R16

DATO:   OUT     UDR,    R16                 ;SE TRANSMITE CARACTER 'R'

LOOP:   SBIS    UCSRA,  UDRE                ;SE ESPERA EL FIN DE TRANSMICION DEL CARACTER
        RJMP    LOOP
DELAY1: LDI     R22,    $2F
        RJMP    DLY1
DLY1:   DEC     R20
        BRNE    DLY1
        DEC     R21
        BRNE    DLY1
        DEC     R22
        BRNE    DLY1
        RJMP    INICIO
```

5 CONCLUSIONES

El conocimiento adquirido al momento de realizar el semáforo fue de gran utilidad al momento de realizar esta práctica, creo que fue mucho mas sencilla de realizar pues solo se genero una subrutina.