Đại học Quốc gia TPHCM
Trường Đại học Khoa học tự nhiên
Khoa Vật lý – Vật lý kỹ thuật
Bộ môn Vật lý Tin học

***

THỰC HÀNH VI ĐIỀU KHIỂN
(PHY10605)

CBHD: Võ Hoàng Thủy Tiên
vhttien@hcmus.edu.vn
0937649914

Huỳnh Quốc Việt
hyqviet@hcmus.edu.vn
0349043204

❖ Lưu ý

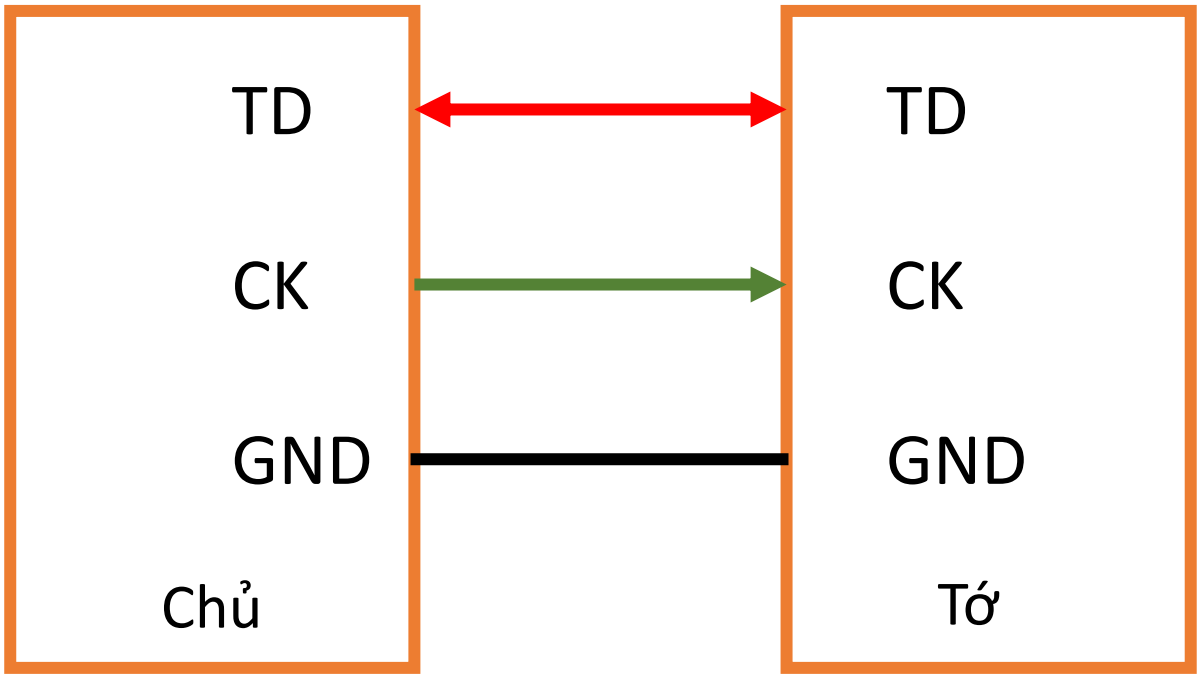
- Nghỉ giữa kì: 16/06/2020
- Nộp cuốn báo cáo + bảng phân công: **03/07/2020**
- Ngày báo cáo: **07/07/2020**

1. Các kiểu truyền dữ liệu

- Truyền dữ liệu nối tiếp đồng bộ và bất đồng bộ (UART synchronouns asynchronouns receiver - transmitter)
- Truyền dữ liệu giữa vi điều khiển với các thiết bị ngoại vi (SPI – Serial Peripheral Interface)
- Truyền dữ liệu hai dây (I2C – Inter-Integrated Circuit)

1.1 Dữ liệu nối tiếp đồng bộ và bất đồng bộ

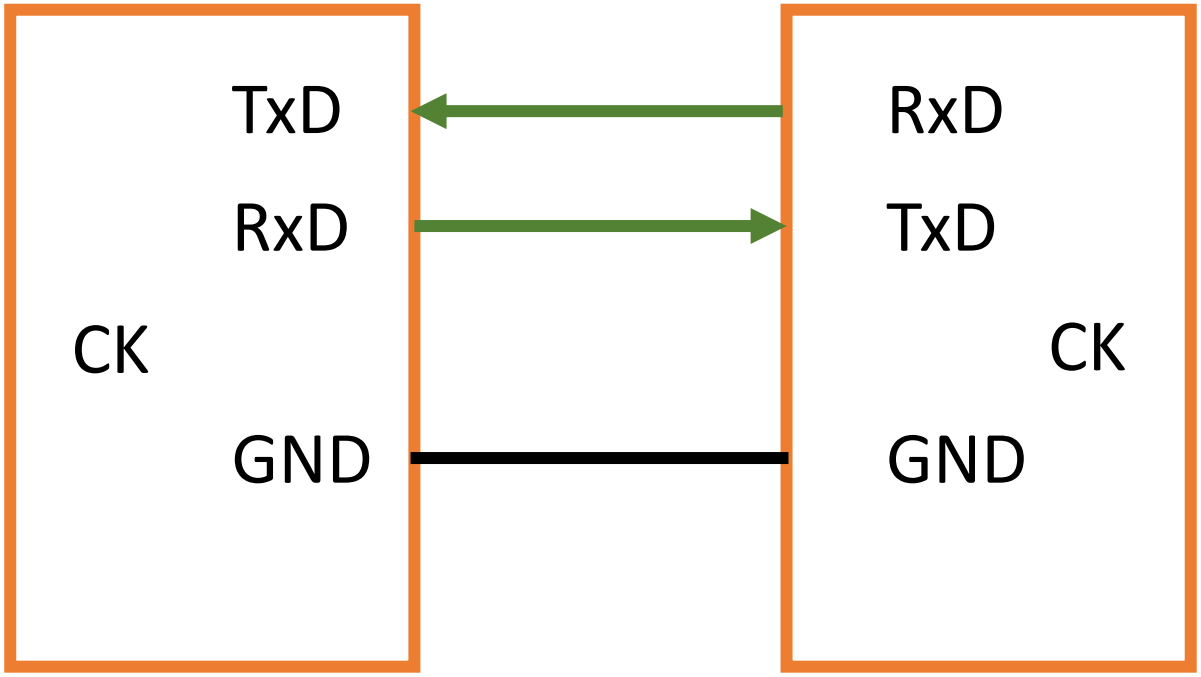
- Truyền dữ liệu nối tiếp đồng bộ: đường truyền dữ liệu (TD) và tín hiệu xung (CK)



Hình 1 Hệ thống truyền đồng bộ

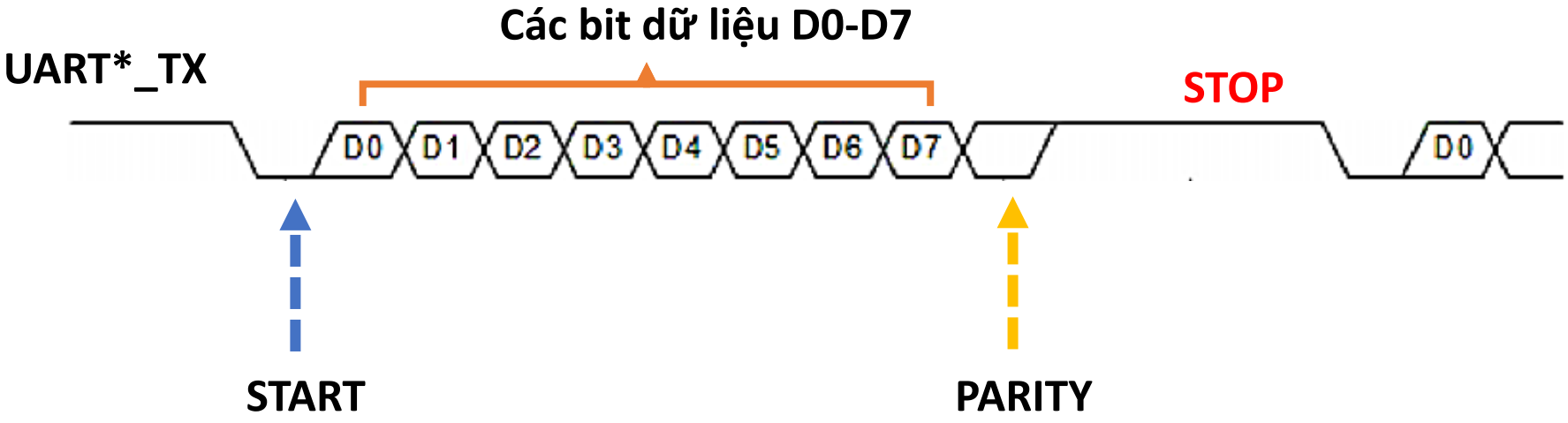
1.1 Dữ liệu nối tiếp đồng bộ và bất đồng bộ

- Truyền dữ liệu nối tiếp bất đồng bộ: là hệ thống ngang cấp, có 2 mạch dao động độc lập nhưng cùng tần số hay tốc độ



Hình 2 Hệ thống truyền bất đồng bộ

1.2 Truyền dữ liệu UART



Baud rate: số bit truyền được trong 1s(cài đặt giống nhau ở truyền và nhận)

Frame(khung truyền): quy định số bit cho mỗi lần truyền

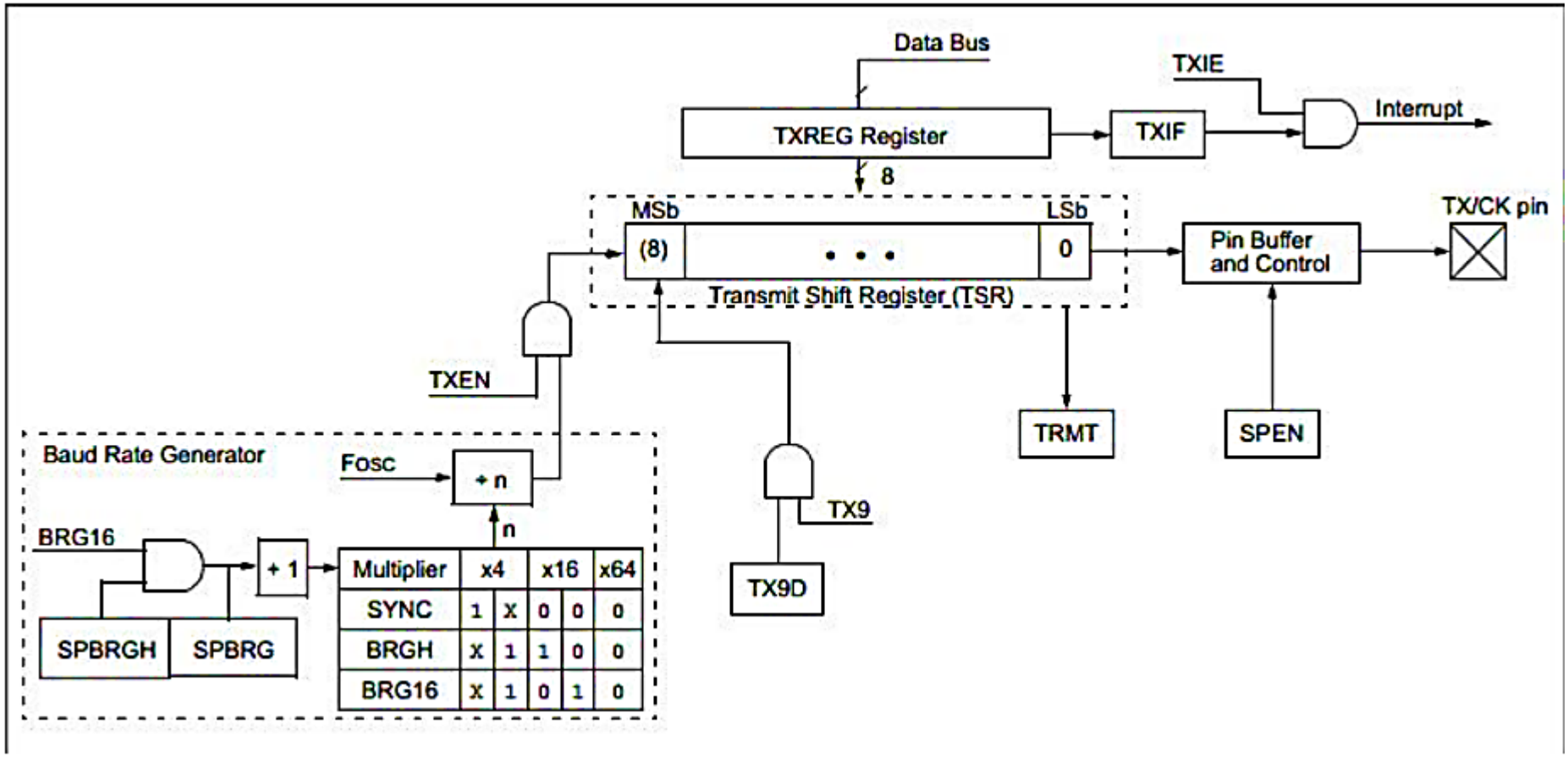
1.3 Tốc độ truyền dữ liệu UART

SYNC	BRG16	BRGH	Chế độ truyền	Công thức
0	0	0	8-bit/BĐB	$F_{osc}/[64 * (n+1)]$
0	0	1	8-bit/BĐB	$F_{osc}/[16 * (n+1)]$
0	1	0	16-bit/BĐB	
0	1	1	16-bit/BĐB	$F_{osc}/[4 * (n+1)]$
1	0	x	8-bit/ĐB	
1	1	x	16-bit/ĐB	

SYNC	BRG16	BRGH	Chế độ truyền	Công thức
0	0	0	8-bit/BĐB	$F_{osc}/[64 * (n+1)]$
0	0	1	8-bit/BĐB	$F_{osc}/[16 * (n+1)]$
0	1	0	16-bit/BĐB	
0	1	1	16-bit/BĐB	$F_{osc}/[4 * (n+1)]$
1	0	x	8-bit/ĐB	
1	1	x	16-bit/ĐB	

Hãy tính giá trị n của cặp thanh ghi để tốc dộ truyền là 9600BAUD, sử dụng thạch anh có tấn số 16MHz, hoạt động bất đồng bộ, bộ phát tốc độ BRG 8-bit

1.4 Khối truyền dữ liệu



1.4 Khối truyền dữ liệu

1. Dữ liệu cần truyền được đặt vào thanh ghi TXREG, baud rate được tạo ra, khi TXEN gán bằng 1 dữ liệu từ thanh ghi TXREG đưa vào thanh ghi TSR đồng thời baud rate tác động đến TSR, đẩy dữ liệu cần truyền ra bộ đệm sau đó xuất ra chân TX.

2. Bit TXIF dùng để báo trạng thái trong thanh ghi TXREG, nếu có dữ liệu trong TXREG thì TXIF =1. Nếu dữ liệu được truyền xuống thanh TSR thì TXIF = 0. Tương tự bit TRMT dùng để báo trạng thái thanh ghi TSR.

1.4 Khối truyền dữ liệu

Các bước cho quá trình truyền dữ liệu:

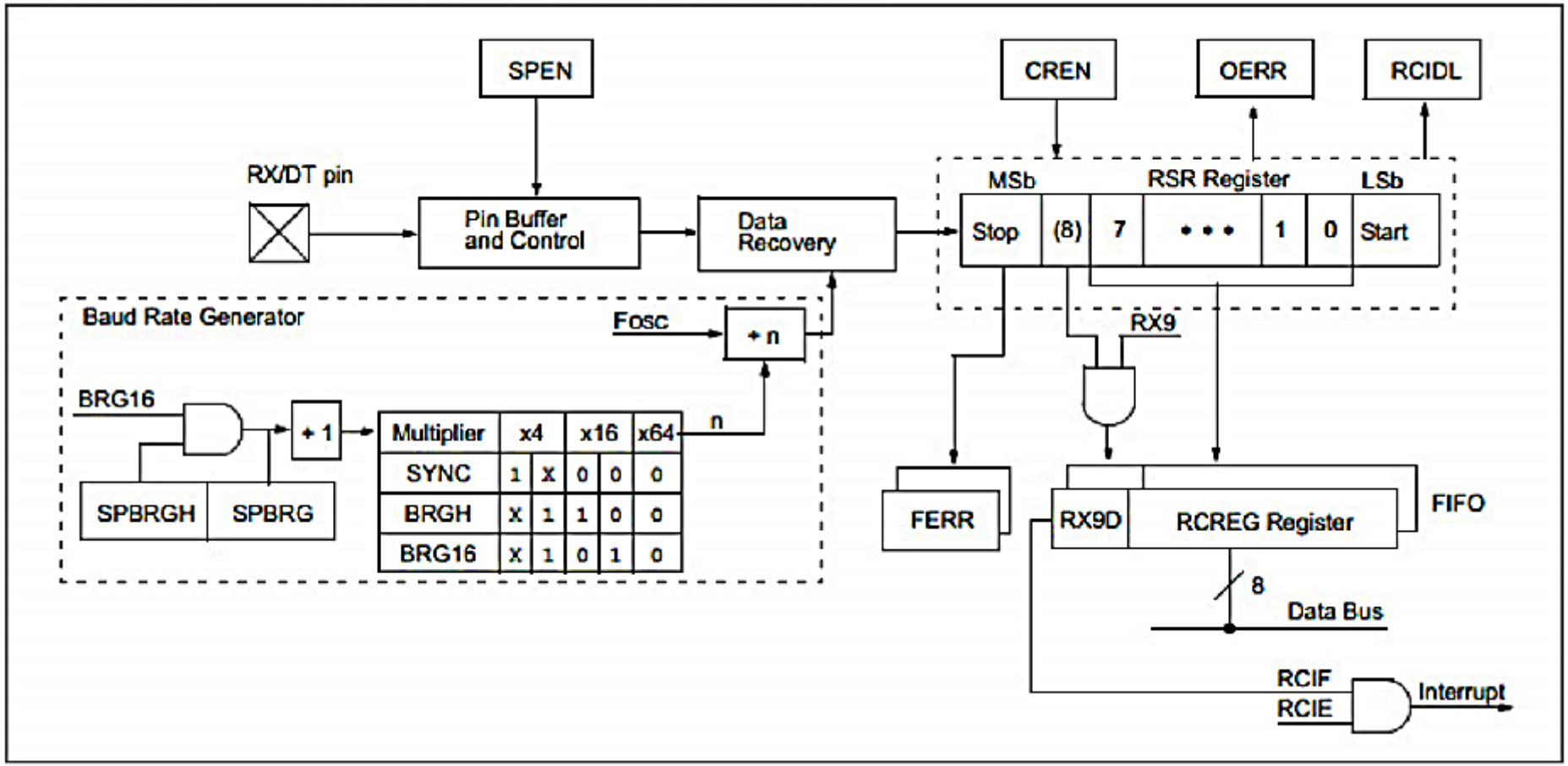
1. Khởi tạo baund rate: ở thanh ghi SPBRG
Cho phép quá trình truyền thông không đồng bộ bằng cách thiết lập SPEN = 1; SYNC = 0;
2. Cho phép truyền dữ liệu bằng cách thiết lập bit TXEN = 1;
3. Khi cần truyền dữ liệu thì cần set dữ liệu đó lên TXREG.

Thanh ghi quy định chế độ truyền.

- **TX9**: Cho phép truyền nhận chế độ 9 bit. **TX9** = 1; // Hoạt động với chế độ 9 bit. **TX9** = 0; // Hoạt động với chế độ 8 bit.
- **TXEN**: Cho phép truyền UART. **TXEN** = 1; // Cho phép. **TXEN** = 0; // Không cho phép
- **SYNC**: Cho phép chế độ đồng bộ. **SYNC** = 1; // Truyền chế độ đồng bộ. **SYNC** = 0; // Truyền chế độ bất đồng bộ.
- **BRGH**: Chọn chế độ baund rate. **BRGH** = 1; // Tốc độ cao (bất đồng bộ). **BRGH** = 0; // Tốc độ thấp (bất đồng bộ).
- **TRMT**: Trạng thái thanh ghi truyền. **TRMT** = 1; // Thanh ghi TSR trống.**TRMT** = 0; // Thanh ghi TSR có dữ liệu.
- **TX9D**: Dữ liệu bit thứ 9 trong chế độ truyền 9 bit.

1.5 Khối nhận dữ liệu

FIGURE 12-2: EUSART RECEIVE BLOCK DIAGRAM



1.5 Khối nhận dữ liệu

Các bước cho quá trình nhận dữ liệu bao gồm:

1. Khởi tạo baund rate: ở thanh ghi SPBRG

Cho phép quá trình truyền thông không đồng bộ bằng cách thiết lập

SPEN = 1; **SYNC** = 0;

2. Cho phép ngắt quá trình nhận dữ liệu **CREN** = 1;

3. Cho phép ngắt toàn cục: **CIE** = 1; **PEIE** = 1;

4. Xử lý các phần khác của chương trình khi có ngắt xảy ra thì xử lý dữ liệu.

1.5 Khối nhận dữ liệu

Thanh ghi quy định chế độ nhận:

- **SPEN**: Khởi tạo cổng nối tiếp. SPEN = 1; // Cho phép cổng nối tiếp. SPEN = 0; // Không cho phép
- **RX9**: Cho phép nhận 9bit. RX9 = 1; // Cho phép nhận 9bit. RX9 = 0; // Nhan 8bit.
- **CREN**: Cho phép nhận liên tục. CREN = 1; // Cho phép. CREN = 0; // Không cho phép
- **ADDEN**: Bit cho phép phát hiện địa chỉ (sử dụng ở chế độ truyền nhận bất đồng bộ 9 bit). ADDEN = 1; // Cho phép phát hiện địa chỉ , cho phép ngắt và tải bộ đệm nhận khi RSR<8> được set.- ADDEN = 0; // Không cho phép phát hiện địa chỉ , tất cả byte được nhận và bit thứ 9 dùng làm bit parity.
- **FERR**: Bit báo lỗi frame. FERR == 1; // Có lỗi. FERR == 0; // Không có lỗi.
- **OERR**: Lỗi OVERRUN. OEER == 1 ; // Có lỗi. OEER == 0; // Không lỗi
- **RX9D**: Lưu dữ liệu nhân của bit thứ 9
- Thanh ghi **TXREG**: Dùng để chứa dữ liệu truyền đi.
- Thanh ghi **RCREG**: Dùng để lưu dữ liệu từ ngoài vào.
- Thanh ghi **SPBRG**: Thiết lập baud rate của PIC

2. Các lệnh trong CCS

Value = getc() nhận một kí tự từ chân nhận dữ liệu

gets() nhận chuỗi kí tự từ chân nhận dữ liệu

putc() đặt một kí tự thông qua chân truyền dữ liệu

puts() đặt chuỗi kí tự thông qua chân truyền dữ liệu

printf() in chuỗi kí tự

kbhit() trả về giá trị true khi nhận được một kí tự

3. Thực hành

Bài tập: Sử dụng vi điều khiển PIC 16f877a giao tiếp UART hiển thị nội dung nhập từ bàn phím lên LCD(16x2)

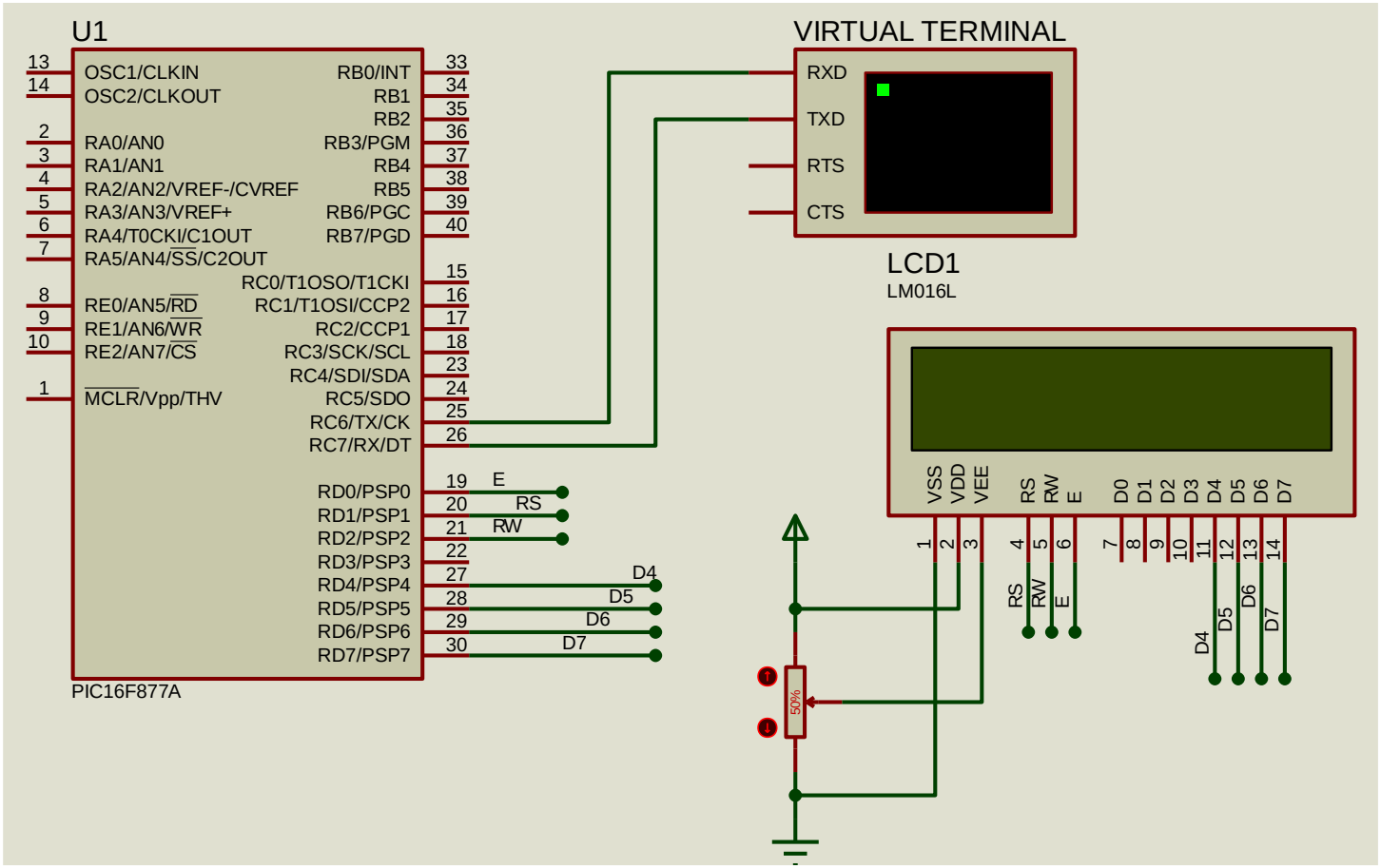
```
#include <VDK5.h>
#include <lcd.c>

#define LCD_DATA4      PIN_D4
#define LCD_DATA5      PIN_D5
#define LCD_DATA6      PIN_D6
#define LCD_DATA7      PIN_D7

#use rs232(baud=9600, parity=N, xmit=PIN_C6, rcv=PIN_C7, bits=8)
```

3. Thực hành

Bài tập: Sử dụng vi điều khiển PIC 16f877a giao tiếp UART hiển thị nội dung nhập từ bàn phím lên LCD(16x2)




TRUONG DAI HOC KHOA HOC TU NHIEN
KHTN
TP. HO CHI MINH


Java
PicC
C++

CẢM ƠN