PORTUL SERIAL

1. Scopul lucrării

Lucrarea urmăreşte cunoaşterea principiilor comunicaţiei seriale şi a modului de realizare a porturilor seriale pentru calculatoarele IBM PC, în particular a circuitelor din familia 16x50 utilizate la aceste calculatoare.

2. Consideraţii teoretice

2.1. Modelul comunicaţiei seriale

Un exemplu de sistem care utilizează comunicaţia serială este ilustrat în Figura 3.1.

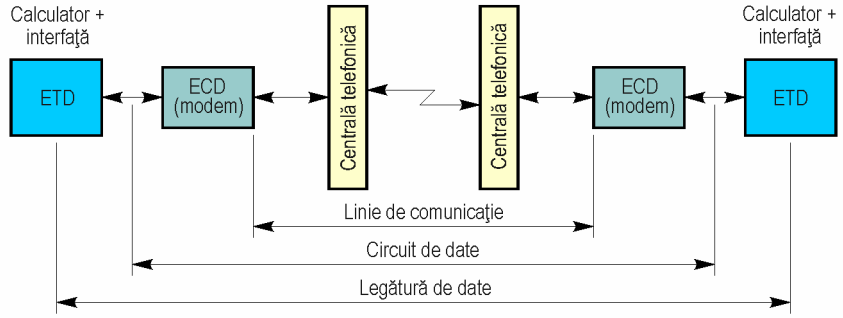

Figura 3.1. Sistem de comunicaţie serială.

Componentele unui sistem de comunicaţie serială sunt următoarele:

1. ETD – *Echipamente terminale de date* (calculatoare, terminale de date). Acestea conţin şi interfeţele seriale sau controlerele de comunicaţie.
2. ECD – *Echipamente pentru comunicaţia de date*. Aceste echipamente se numesc modemuri şi permit calculatorului să transmită informaţii printr-o linie telefonică analogică. Funcţiile principale realizate de un modem sunt următoarele:
   - Conversia digital/analogică a informaţiilor din calculator şi conversia analog/digitală a semnalelor de pe linia telefonică analogică.
   - Modularea/demodularea unui semnal purtător. La transmisie, modemul suprapune (modulează) semnalele digitale ale calculatorului pe frecvenţa purtătoare a liniei telefonice. La recepţie, modemul extrage (demodulează) informaţiile transportate de semnalul purtător şi le transferă calculatorului.
3. *Linia de comunicaţie* reprezintă o linie fizică sau o linie telefonică. Linia telefonică poate fi, la rândul ei, o linie comutată (conectată la o centrală telefonică) sau o linie închiriată (dedicată).
4. *Circuitul de date* cuprinde porţiunea dintre două echipamente terminale de date, deci, modemurile şi linia de comunicaţie. În cazul unor distanţe reduse, este posibilă comunicaţia serială directă între două echipamente terminale de date prin linii fizice, fără utilizarea unor modemuri. În acest caz, circuitul de date este reprezentat de aceste linii.

5. *Legătura de date* conține circuitul de date şi interfețele seriale ale echipamentelor terminale de date.

Numărul echipamentelor interconectate printr-o legătură serială poate fi de două (într-o *legătură punct la punct*) sau mai mult de două (într-o *legătură multi-punct*).

2.2. Parametrii comunicației seriale

Viteza de transmisie (numită şi *debit binar*) este măsurată în biți/s:

$$D = \frac{1}{T} \text{ [biți/s]}$$

unde T este perioada de timp necesară pentru transmiterea unui bit.

Modemul reprezintă semnalele de date prin diferite stări electrice, în funcție de tipul de modulație pe care îl utilizează: frecvență, amplitudine, fază. Fiecare stare electrică este menținută la ieşirea modemului pentru un interval de timp numit *interval de modulație* (Δ). *Viteza de modulație* este inversul intervalului de modulație, reprezentând numărul schimbărilor pe secundă ale stării electrice a modemului:

$$V_m = \frac{1}{\Delta} \text{ [baud]}$$

Unitatea de măsură a vitezei de modulație este *baud*, după numele inginerului şi telegrafistului francez Jean-Maurice Baudot. Relația dintre viteza de transmisie D şi viteza de modulație V_m este:

$$D = V_m \cdot \log_2 n \text{ [biți/s]}$$

unde n este numărul stărilor electrice distincte ale modemului. În particular, dacă există doar două stări electrice distincte ale modemului, viteza de transmisie este egală cu viteza de modulație. În general însă, există un număr mai mare de stări electrice ale modemului, astfel încât viteza de transmisie este un multiplu al vitezei de modulație.

Deseori, se confundă viteza de modulație cu viteza de transmise (debitul binar). Viteza de modulație (exprimată în baud) este viteza cu care se modifică stările electrice ale modemului. De exemplu, dacă se utilizează modulația în frecvență, iar frecvența semnalului purtător se poate modifica de către modem de 2.400 de ori pe secundă, viteza de modulație este de 2.400 baud. Primele modemuri codificau un bit de 0 printr-o anumită frecvență şi un bit de 1 printr-o altă frecvență. În acest caz particular, viteza de modulație are aceeaşi valoare cu viteza de transmisie. Modemurile actuale codifică însă mai mulți biți de informație printr-o stare electrică. De exemplu, dacă modemul codifică 4 biți de informație printr-o anumită frecvență, pentru exemplul anterior viteza de transmisie va fi de $4 \times 2.400 = 9.600$ biți/s.

2.3. Tipuri de comunicație serială

Din punctul de vedere al *direcției de transfer*, există următoarele tipuri de comunicație serială:

- Simplex;
- Semiduplex;
- Duplex.

În cazul comunicației *simplex*, datele sunt transferate întotdeauna în aceeaşi direcție, de la echipamentul transmițător la cel receptor. La comunicația *semiduplex*, fiecare echipament terminal de date funcționează alternativ ca transmițător, iar apoi ca receptor. Pentru o asemenea conexiune, este suficientă o singură linie de transmisie (două fire de legătură). Într-o comunicație *duplex* (numită şi duplex integral), datele se transferă simultan în ambele direcții. Primele conexiuni duplex necesitau două linii de transmisie (patru fire de legătură), dar conexiunile actuale necesită o singură linie.

Din punctul de vedere al *sincronizării* dintre transmițător şi receptor, există două tipuri de comunicație serială:

- Asincronă;
- Sincronă.

2.3.1. Comunicația asincronă

Pentru a asigura sincronizarea dintre transmițător şi receptor, fiecare caracter transmis este precedat de un bit de START, având valoarea logică 0, şi este urmat de cel puțin un bit de STOP, cu valoarea logică 1. Biții de START şi de STOP încadrează deci fiecare caracter transmis, caracterul transmis între cei doi biți reprezentând un *cadru* de date. Un asemenea cadru reprezintă informația digitală de bază într-un sistem de comunicație serială. În cazul comunicației asincrone, intervalul de timp între transmisia a două caractere succesive este variabil, pe durata acestui interval linia de comunicație fiind în starea 1 logic ("*mark*"). Acest mod de comunicație este numit şi *start-stop*.

Sincronizarea la nivel de bit se realizează cu ajutorul semnalelor de ceas locale cu aceeaşi frecvență. Atunci când un receptor detectează începutul unui caracter indicat prin bitul de START, porneşte un oscilator de ceas local, care permite eşantionarea corectă a biților individuali ai caracterului. Eşantionarea biților se realizează aproximativ la jumătatea intervalului corespunzător fiecărui bit.

În Figura 3.2 se ilustrează transmisia caracterului cu codul ASCII 61h. După bitul de START, având durata T corespunzătoare unui bit, transmisia caracterului începe cu bitul cel mai puțin semnificativ b_0. După transmisia bitului cel mai semnificativ b_7, se transmite un bit de paritate p, în exemplul ilustrat paritatea fiind impară. Acest bit de paritate este opțional, iar în cazul în care se adaugă la caracterul transmis, paritatea poate fi selectată ca fiind pară sau impară. Există şi posibilitatea ca bitul de paritate să fie întotdeauna 0 sau 1, indiferent de paritatea efectivă a caracterului. La sfârşitul caracterului, în exemplul ilustrat se transmit doi biți de STOP s_1 şi s_2, după care linia rămâne în starea 1 logic un timp variabil. Acest timp corespunde unui interval de pauză.

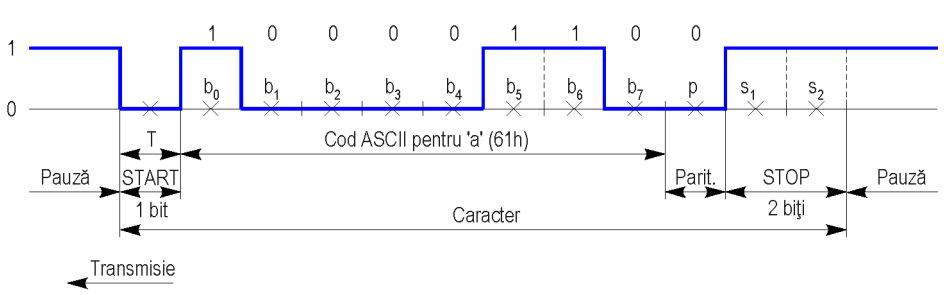

Figura 3.2. Comunicație asincronă.

În cazul comunicației asincrone, sincronizarea la nivel de bit este asigurată numai pe durata transmisiei efective a fiecărui caracter. O asemenea comunicație este orientată pe caractere individuale şi are dezavantajul că necesită informații suplimentare în proporție de cel puțin 25% pentru identificarea fiecărui caracter.

2.3.2. Comunicația sincronă

În cazul comunicației sincrone, un cadru nu conține un singur caracter, ci un *bloc* de caractere sau un *mesaj*. Sincronizarea la nivel de bit trebuie asigurată permanent, nu numai în timpul transmisiei propriu-zise, ci şi în intervalele de pauză. De aceea, timpul este divizat în mod continuu în intervale elementare la transmițător, intervale care trebuie regăsite apoi la receptor. Aceasta pune anumite probleme. Dacă ceasul local al receptorului are o frecvență care diferă într-o anumită măsură de frecvența transmițătorului, vor apare erori la recunoaşterea caracterelor, din cauza lungimii blocurilor de caractere.

Pentru a se evita aceste erori, ceasul receptorului trebuie resincronizat frecvent cu cel al transmițătorului. Aceasta se poate realiza dacă se asigură că există suficiente tranziții de la 1 la 0 şi de la 0 la 1 în mesajul transmis. Dacă datele de transmis constau din şiruri lungi de 1 sau de 0, trebuie inserate tranziții suficiente pentru resincronizarea ceasurilor. Asemenea tehnici sunt dificil de

implementat, astfel încât se utilizează de obicei o tehnică numită comunicație asincronă sincronizată (numită în mod simplu comunicație sincronă).

Acest tip de comunicație este caracterizat de faptul că, deşi mesajul este transmis într-un mod sincron, nu există o sincronizare în intervalul de timp dintre două mesaje. Informația este transmisă sub forma unor blocuri de caractere sau a unor biți succesivi, fără biți de START şi STOP. Pentru ajustarea oscilatorului local la începutul unui mesaj, fiecare mesaj este precedat de un număr de caractere speciale de sincronizare (de exemplu, caracterul SYN, cu valoarea 16h). Pentru menținerea sincronizării, se pot insera caractere de sincronizare suplimentare în mesajul transmis, la anumite intervale de timp.

La receptor există trei nivele de sincronizare:

- Sincronizare la nivel de bit, realizată prin bucle cu calare de fază PLL (*Phase–Locked Loop*)[1] pe baza tranzițiilor existente în semnalul recepționat;
- Sincronizare la nivel de caracter, realizată prin recunoaşterea unor caractere de sincronizare;
- Sincronizare la nivel de bloc sau mesaj, care depinde de protocolul de date utilizat.

2.4. Standardul RS-232C

Specificațiile electrice ale portului serial utilizat la calculatoarele IBM PC au fost definite în standardul RS-232C (*Reference Standard No. 232, Revision C*), elaborat în anul 1969 de către Comitetul de Standarde din SUA, cunoscut azi sub numele de Asociația Industriei Electronice (EIA – *Electronics Industry Association*). Standardul a fost elaborat pentru comunicația digitală între un calculator şi un terminal aflat la distanță sau între două terminale, fără utilizarea unui calculator. Terminalele erau conectate prin linii telefonice, astfel încât erau necesare modemuri la ambele capete ale liniei de comunicație.

Standardul RS-232C a suferit diferite modificări, fiind elaborate mai multe revizii ale acestuia. Astfel, în anul 1987 a fost elaborată o nouă revizie a standardului, numită EIA RS-232D. În anul 1991, EIA împreună cu Asociația Industriei de Telecomunicații (TIA – *Telecommunications Industry Association*) a elaborat revizia E a standardului (EIA/TIA RS-232E). Revizia curentă este EIA RS-232F, publicată în anul 1997. Totuşi, indiferent de revizia acestuia, standardul este numit de cele mai multe ori RS-232C sau RS-232.

În Europa, versiunea echivalentă standardului RS-232C este V.24, elaborată de organizația CCITT (*Comité Consultatif International pour Téléphonie et Télégraphie*). Denumirea acestei organizații a fost schimbată la începutul anilor 1990 în *International Telecommunications Union* (ITU). Ambele standarde specifică semnalele utilizate pentru comunicație, nivelele de tensiune, protocolul utilizat pentru controlul fluxului de date şi conectorii interfeței seriale.

Standardul RS-232C defineşte atât o comunicație asincronă, cât şi una sincronă. Nu sunt definite elemente cum sunt codificarea caracterelor (ASCII, Baudot, EBCDIC), încadrarea caracterelor (numărul de biți/caracter, numărul biților de STOP, paritatea) şi nici vitezele de comunicație, deşi standardul este destinat pentru viteze mai mici de 20.000 biți/s. Echipamentele actuale permit însă viteze superioare de comunicație, utilizând nivele de tensiune care sunt compatibile cu cele specificate de standard. Porturile seriale ale calculatoarelor permit de obicei selecția uneia din următoarele viteze de comunicație: 150, 300, 600, 1.200, 2.400, 4.800, 9.600, 19.200, 38.400, 57.600 şi 115.200 biți/s.

O legătură de bază RS-232C necesită doar trei conexiuni: una pentru transmisie, una pentru recepție şi una pentru masa electrică comună. Cele mai multe legături seriale utilizează însă şi semnale pentru controlul fluxului de date.

Spre deosebire de alte tipuri de comunicație serială, care sunt diferențiale[2], comunicația RS-232C este una obişnuită, utilizând câte un fir pentru fiecare semnal. Deşi astfel se simplifică

[1] Un circuit PLL reprezintă un sistem de control în buclă închisă a frecvenței unui oscilator. Funcționarea circuitului PLL se bazează pe detectarea diferenței de fază între semnalele de intrare şi de ieşire ale oscilatorului controlat.

[2] O comunicație diferențială utilizează câte o pereche de fire pentru fiecare semnal. Exemple de comunicații diferențiale sunt cele specificate de standardele RS-422 sau RS-485, dar şi cele ale magistralelor USB sau IEEE 1394.

circuitele necesare interfeței, în acelaşi timp se reduc şi distanța maximă de comunicație în cazul unei legături directe, fără utilizarea modemurilor. Standardul RS-232C specifică o distanță maximă de 15 m. Se pot obține distanțe mai mari dacă se utilizează viteze de comunicație mai reduse.

Tensiunile electrice specificate de standardul RS-232C sunt următoarele:

- Valoarea logică 0 corespunde unei tensiuni pozitive între +3 V şi +25 V;
- Valoarea logică 1 corespunde unei tensiuni negative între –3 V şi –25 V.

2.5. Semnalele interfeței seriale

Interfața serială utilizează atât semnale pentru transmisia şi recepția datelor, cât şi semnale pentru controlul fluxului de date între două echipamente. În Figura 3.3 se prezintă denumirea semnalelor interfeței seriale conform standardului RS-232C. Legătura ilustrată este cea pentru care a fost conceput inițial portul serial, şi anume, conectarea unui modem la calculator. Cifrele indică numărul pinilor conectorilor DB-25 utilizați în cazul unei asemenea legături prin modemuri, iar cifrele din paranteze reprezintă semnalele conform standardului V.24.

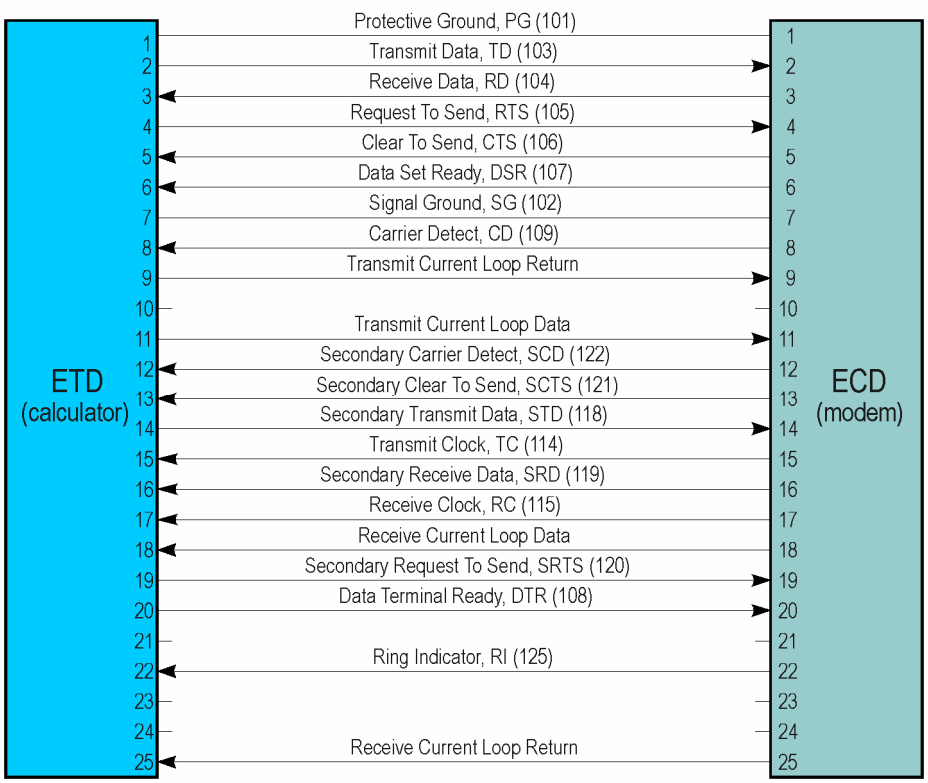

Figura 3.3. Semnalele interfeței seriale.

Principalele semnale sunt descrise în continuare.

Transmit Data, TD (*Transmisie Date*)

Datele sunt transmise serial pe această linie de către calculator. După bitul de start, se transmite bitul cel mai puțin semnificativ al unui caracter. În general, pentru transmisie este necesar ca semnalele *RTS*, *CTS*, *DTR* şi *DSR* să fie active. Aceste semnale sunt activate în cadrul secvenței de stabilire a legăturii cu modemul.

Receive Data, RD (*Recepție date*)

Această linie este utilizată de calculator pentru recepția datelor de la modem sau de la un echipament extern.

Data Terminal Ready, DTR (*Terminal de date operațional*)

Atunci când calculatorul este operațional şi pregătit pentru comunicația de date, activează semnalul *DTR*. Modemul va răspunde la semnalul *DTR* prin semnalul *DSR*.

Data Set Ready, DSR (*Modem operațional*)

Atunci când modemul sau echipamentul extern este operațional şi pregătit pentru comunicația de date, activează semnalul *DSR*. Semnalul *DSR* este activat de modem ca răspuns la activarea semnalului *DTR* de către calculator. Calculatorul va transmite date către modem doar în cazul în care semnalul *DSR* este activ.

Request To Send, RTS (*Cerere de emisie*)

Atunci când calculatorul este pregătit pentru transmisia datelor, activează semnalul *RTS*. Acest semnal indică modemului faptul că poate transmite date către calculator. Un semnal *RTS* inactiv va preveni modemul de a transmite date către calculator. Aceasta permite calculatorului să controleze fluxul datelor transmise de modem. Răspunsul la semnalul *RTS* se recepționează de calculator pe linia *CTS*.

Clear To Send, CTS (*Gata de emisie*)

Prin activarea acestui semnal, modemul sau echipamentul extern indică faptul că este pregătit pentru recepția datelor de la calculator. Semnalul *CTS* este activat de modem ca răspuns la activarea semnalului *RTS* de către calculator. Un semnal *CTS* inactiv va preveni calculatorul de a transmite date către modem. Aceasta permite modemului să controleze fluxul datelor transmise de calculator.

Carrier Detect, CD (*Detectare purtătoare de semnal*)

Prin activarea acestui semnal, modemul semnalează calculatorului faptul că a detectat semnalul purtător al altui modem pe linia telefonică, deci, există o conexiune cu un modem aflat la distanță. Într-o legătură serială fără modemuri, semnalul *CD* activ indică faptul că este posibilă comunicația cu un echipament de la celălalt capăt. De multe ori, acest semnal este ignorat de calculator.

Ring Indicator, RI (*Indicator de apel*)

Atunci când modemul detectează pe linia telefonică semnalul de apel de la un alt modem, activează semnalul *RI*. Acest semnal permite programului care rulează pe calculator să răspundă în mod automat la un apel telefonic de la distanță.

Transmit Clock, TC (*Ceas pentru transmisie*)

Reprezintă semnalul de ceas pentru transmisie furnizat calculatorului de către modem în cazul unei comunicații sincrone. Nu este utilizat pentru comunicația asincronă.

Receive Clock, RC (*Ceas pentru recepție*)

Reprezintă semnalul de ceas pentru recepție furnizat calculatorului de către modem în cazul unei comunicații sincrone. Nu este utilizat pentru comunicația asincronă.

Transmit Current Loop Data
Transmit Current Loop Return
Receive Current Loop Data
Receive Current Loop Return

Aceste semnale permit comunicația între echipamente aflate la o distanță apropiată, fără utilizarea unor modemuri. Metoda utilizată se numeşte transmisie prin *buclă de curent*. Nivelul logic 0 este indicat printr-un curent de 20 mA, iar nivelul logic 1 este indicat prin absența acestui curent. Conectarea unui echipament care utilizează comunicația în buclă de curent la un port serial compatibil RS-232C necesită un translator al nivelului de tensiune.

Secondary Transmit Data, STD
Secondary Receive Data, SRD

Reprezintă semnalele de date pentru o a doua legătură serială. Deşi teoretic sunt posibile două legături seriale duplex printr-un singur cablu, în practică a doua legătură este implementată foarte rar.

Secondary Request To Send, SRTS
Secondary Clear To Send, SCTS
Secondary Carrier Detect, SCD

Reprezintă semnalele *RTS*, *CTS*, respectiv *CD* pentru a doua legătură serială.

Secvenţa de stabilire a legăturii între calculator şi modem constă pe scurt din următoarele operaţii: programul activează semnalul *DTR* şi aşteaptă răspunsul prin semnalul *DSR* de la modem. În continuare programul activează semnalul *RTS* şi aşteaptă răspunsul prin semnalul *CTS* de la modem.

2.6. Controlul fluxului de date

Pentru a fi posibilă comunicaţia între dispozitive cu viteze diferite, proiectanţii interfeţei seriale au prevăzut semnale speciale destinate controlului fluxului de date. Aceste semnale permit oprirea şi apoi reluarea transmiterii datelor de către un echipament la cererea celuilalt echipament cu care se realizează comunicaţia serială. Pe lângă această *metodă hardware* pentru controlul fluxului de date, există şi o *metodă software* de control, bazată pe transmiterea unor caractere speciale între cele două echipamente. Atunci când echipamentul receptor (de exemplu, o imprimantă) nu mai poate primi date din cauza umplerii bufferului acestuia, transmite echipamentului transmiţător (de exemplu, calculatorului) un anumit caracter de control. Atunci când echipamentul receptor poate primi noi date, transmite un alt caracter de control prin care indică echipamentului transmiţător că poate relua transmiterea datelor.

Metoda de control care va fi utilizată de calculator poate fi selectată de obicei prin intermediul driverului controlerului serial. Unele programe pot utiliza în mod implicit o anumită metodă. În cazul perifericelor, metoda de control poate fi selectată fie prin program, fie cu ajutorul unui comutator. Este important ca atât calculatorul, cât şi perifericul să utilizeze aceeaşi metodă de control pentru a evita pierderea datelor.

2.6.1. Metoda de control hardware

Această metodă presupune utilizarea unui protocol de comunicaţie cu ajutorul semnalelor de control ale interfeţei seriale. Protocolul utilizat se bazează pe comunicaţia serială prin intermediul unor modemuri şi a unei linii telefonice, pentru care a fost elaborată interfaţa serială originală. Acest protocol implică stabilirea conexiunii între două modemuri prin linia telefonică şi menţinerea fluxului de date dintre acestea în timpul în care conexiunea este activă. Etapele acestui protocol sunt descrise în continuare. Într-o formă simplificată, acest protocol este utilizat şi în cazul comunicaţiei seriale directe între două echipamente, fără utilizarea unor modemuri şi a unei linii telefonice.

1. Atunci când un modem aflat la distanţă doreşte stabilirea legăturii cu modemul local, transmite pe linia telefonică semnalul de apel. Acest semnal este detectat de către modemul local, care activează semnalul *RI* pentru a informa calculatorul local asupra existenţei unui apel telefonic.
2. La detectarea activării semnalului *RI*, pe calculatorul local se lansează în execuţie un program de comunicaţie. Acest program indică disponibilitatea calculatorului de a începe comunicaţia prin activarea semnalului *DTR*.
3. Atunci când modemul local sesizează faptul că terminalul de date (calculatorul) este pregătit, răspunde la apelul telefonic şi aşteaptă activarea semnalului purtător de către modemul aflat la distanţă. La detectarea semnalului purtător, modemul local activează semnalul *CD*.

4. Modemul local negociază cu modemul aflat la distanță o conexiune cu anumiți parametri. De exemplu, cele două modemuri pot determina viteza optimă de comunicație în funcție de calitatea legăturii telefonice. După această negociere, modemul local activează semnalul *DSR*.
5. La sesizarea activării semnalului *DSR*, programul de pe calculatorul local activează semnalul *RTS* pentru a indica modemului că poate transmite date către calculator.
6. La sesizarea activării semnalului *RTS*, modemul activează semnalul *CTS* pentru a indica faptul că este pregătit pentru recepția datelor de la calculator.
7. În continuare se realizează transferul datelor în ambele sensuri între echipamentele aflate la distanță, prin liniile *TD* şi *RD*.
8. Deoarece viteza liniei telefonice este mai redusă decât cea a legăturii dintre calculator şi modemul local, bufferul acestui modem se va umple. Modemul local solicită calculatorului oprirea transmiterii datelor prin dezactivarea semnalului *CTS*. La golirea bufferului, modemul reactivează semnalul *CTS*.
9. În cazul în care calculatorul nu mai poate primi date de la modem, dezactivează semnalul *RTS*. Atunci când poate primi din nou date de la modem, calculatorul reactivează semnalul *RTS*.
10. La încheierea sesiunii de comunicație, semnalul purtător va fi dezactivat, iar modemul local dezactivează semnalele *CD*, *CTS* şi *DSR*.
11. La sesizarea dezactivării semnalului *CD*, calculatorul local dezactivează semnalele *RTS* şi *DTR*.

Din protocolul descris mai sus, rezultă următoarele:

- Calculatorul trebuie să sesizeze activarea semnalelor *DSR* şi *CTS* înainte de a transmite date către modem. Dezactivarea oricăruia din cele două semnale va opri, de obicei, fluxul de date de la calculator.
- Modemul trebuie să sesizeze activarea semnalelor *DTR* şi *RTS* înainte de a transmite date pe linia serială sau către calculator. Dezactivarea semnalului *DTR* va opri transmiterea datelor pe linia serială, iar dezactivarea semnalului *RTS* va opri transmiterea datelor către calculator.

Starea semnalului *CD* nu este interpretată de toate sistemele de comunicație serială. La anumite sisteme, semnalul *CD* trebuie să fie activat înainte ca terminalul de date să înceapă transmiterea datelor. La alte sisteme, starea semnalului *CD* este ignorată.

2.6.2. Metoda de control software

Metoda software pentru controlul fluxului de date presupune transmiterea unor caractere de control între cele două echipamente. De exemplu, perifericul va transmite un anumit caracter de control pentru a indica faptul că nu mai poate primi date de la calculator şi va transmite un alt caracter de control pentru a indica faptul că transmiterea datelor poate fi reluată de calculator. Există două variante ale acestei metode. La prima variantă, se transmit caracterele de control XON/XOFF, iar la a doua se transmit caracterele de control ETX/ACK.

În cazul variantei XON/XOFF, perifericul transmite caracterul XOFF pentru a indica faptul că bufferul său este plin şi transmiterea datelor trebuie oprită de calculator. Acest caracter mai este numit DC1 (*Device Control* 1) şi are codul ASCII 13h, fiind echivalent cu caracterul Ctrl-S. Caracterul poate fi introdus şi de utilizator la anumite programe de comunicație pentru a opri transmiterea datelor de către echipamentul cu care este conectat calculatorul. Atunci când perifericul este pregătit pentru a primi noi date, transmite calculatorului caracterul XON. Acest caracter mai este numit DC3 (*Device Control* 3) şi are codul ASCII 11h, fiind echivalent cu caracterul Ctrl-Q. La anumite programe de comunicație, introducerea caracterului Ctrl-Q anulează efectul caracterului Ctrl-S.

În cazul variantei ETX/ACK, transmiterea caracterului ETX (*End of TeXt*) de către periferic indică faptul că transmiterea datelor trebuie oprită de calculator. Acest caracter are codul ASCII 03h şi este echivalent cu caracterul Ctrl-C. Transmiterea caracterului ACK (*ACKnowledge*) indică

posibilitatea reluării transmiterii datelor de către calculator. Acest caracter are codul ASCII 06h şi este echivalent cu caracterul Ctrl-F.

2.7. Conectori

Porturile seriale pot utiliza două tipuri de conectori. Conectorul DB-25 cu 25 de pini a fost utilizat la calculatoarele din generațiile anterioare. La calculatoarele actuale se utilizează conectorul DB-9 cu 9 pini. Pentru porturile seriale ale calculatoarelor se utilizează conectori tată, iar pentru porturile seriale ale echipamentelor periferice se utilizează conectori mamă.

Conectorul DB-25 al portului serial are o formă similară cu conectorul DB-25 al portului paralel. Portul serial care utilizează un conector DB-25 se poate deosebi de portul paralel prin faptul că pentru portul serial se utilizează un conector tată, în timp ce pentru portul paralel se utilizează un conector mamă. Figura 3.4 ilustrează conectorul DB-25 al portului serial.

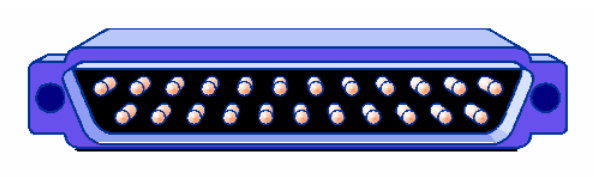

Figura 3.4. Conectorul DB-25 utilizat pentru porturile seriale ale calculatoarelor IBM PC din generațiile anterioare.

Din cele 25 de semnale ale conectorului DB-25, se utilizează cel mult 10 semnale pentru o conexiune serială obişnuită. Tabelul 3.1 indică numele acestor semnale şi asignarea lor la pinii conectorului DB-25.

Tabelul 3.1. Asignarea semnalelor la pinii conectorului DB-25 al portului serial.

Pin	Semnal	Semnificație	← In → Out
1	PG	Protective Ground	
2	TD	Transmit Data	→
3	RD	Receive Data	←
4	RTS	Request To Send	→
5	CTS	Clear To Send	←
6	DSR	Data Set Ready	←
7	SG	Signal Ground	
8	CD	Carrier Detect	←
20	DTR	Data Terminal Ready	→
22	RI	Ring Indicator	←

Pentru a se reduce spațiul ocupat de conectorul portului serial, conectorul DB-25 a fost înlocuit cu un conector de dimensiuni mai reduse, conectorul cu 9 pini DB-9 (Figura 3.5).

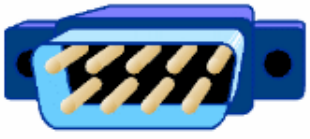

Figura 3.5. Conectorul DB-9 utilizat pentru porturile seriale ale calculatoarelor IBM PC.

Tabelul 3.2 indică asignarea semnalelor portului serial la pinii conectorului DB-9.

Tabelul 3.2. Asignarea semnalelor la pinii conectorului DB-9 al portului serial.

Pin	Semnal	Semnificație	← In → Out
1	CD	Carrier Detect	←
2	RD	Receive Data	←
3	TD	Transmit Data	→
4	DTR	Data Terminal Ready	→
5	SG	Signal Ground	
6	DSR	Data Set Ready	←
7	RTS	Request To Send	→
8	CTS	Clear To Send	←
9	RI	Ring Indicator	←

2.8. Cabluri

Există diferite variante constructive de cabluri care se pot utiliza pentru comunicația serială. Pentru viteze de comunicație şi lungimi reduse, se pot utiliza cabluri obişnuite, care nu sunt ecranate. Pentru a reduce interferențele cu alte echipamente, trebuie utilizate cabluri ecranate care dispun de un înveliş sub forma unei folii de aluminiu. În mod ideal, ecranul cablului trebuie conectat la masa de protecție a conectorului, dacă acesta este de tip DB-25. La conectorul DB-9 nu există un pin pentru masa de protecție. În cazul utilizării conectorilor de acest tip, ecranul cablului se poate conecta la masa electrică.

Observații

- În cazul cablului serial care utilizează conectori DB-25, masa electrică sau masa de semnal *SG* (*Signal Ground*) este separată de masa mecanică sau masa de protecție *PG* (*Protective Ground*). Masa de protecție este conectată direct la carcasa conectorului (şi a echipamentului), având un rol de protecție. Prin realizarea acestei conexiuni, carcasele metalice ale celor două echipamente conectate prin cablul serial se vor afla la acelaşi potențial, evitându-se formarea unor diferențe de potențial între cele două echipamente care pot fi periculoase pentru acestea. Deseori, conexiunea mesei de protecție lipseşte din cablurile seriale.
- Masa de protecție *PG* nu trebuie conectată niciodată la masa electrică *SG*.

Semnalele interfeței seriale au fost prevăzute în ideea conectării unui echipament terminal de date (ETD) la un echipament pentru comunicația de date (ECD). Atunci când se conectează două asemenea echipamente, de exemplu, un calculator cu un modem, care dispun de conectori de acelaşi tip (de exemplu, DB-25), este necesar un cablu care conectează pinii cu acelaşi număr ai conectorilor de la cele două capete. Un asemenea cablu este numit *direct*. Dacă se conectează două echipamente cu conectori diferiți, este necesar un cablu *adaptor*. Dacă se conectează două echipamente terminale de date, de exemplu, două calculatoare, datele transmise pe pinul *TD* al unui echipament trebuie recepționate pe pinul *RD* al celuilalt echipament. De aceea, conexiunile acestor pini vor fi inversate la cele două capete ale cablului, un asemenea cablu fiind numit cablu *inversor*.

2.8.1. Cabluri directe

Un cablu direct reprezintă o conexiune a unui pin corespunzător unui anumit semnal de la un capăt al legăturii cu pinul corespunzător aceluiaşi semnal de la celălalt capăt al legăturii. În mod obişnuit, se utilizează doar o parte a semnalelor interfeței seriale. Chiar în cazul în care fluxul de date este controlat prin metoda hardware, doar un număr de 9 semnale trebuie conectate între un calculator şi un modem, presupunând o comunicație asincronă. De exemplu, dacă se utilizează conectori DB-25, pinii care trebuie conectați sunt: 2, 3, 4, 5, 6, 7, 8, 20 şi 22. În cazul unei comunicații sincrone, sunt

necesare două conexiuni suplimentare, reprezentând semnalele de ceas pentru transmisie şi recepţie care sunt generate de modemul sincron. Dacă se utilizează conectori DB-9, toţi pinii trebuie conectaţi, cu excepţia pinului 9 (semnalul *RI*).

Figura 3.6 ilustrează conexiunile necesare pentru un cablu serial în cazul conectării unui modem sincron la un calculator, presupunând că se utilizează conectori DB-25 la ambele echipamente.

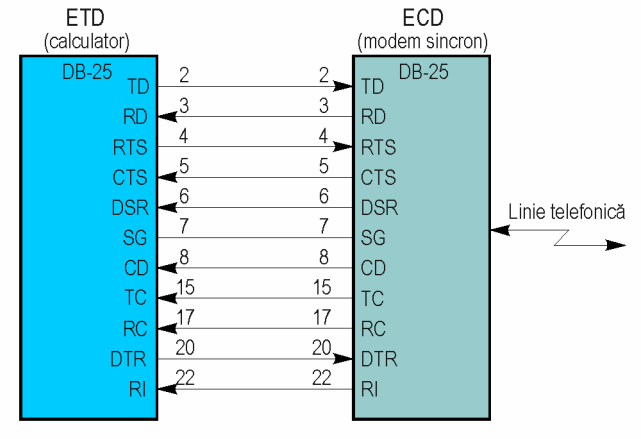

Figura 3.6. Conectarea unui modem sincron la un calculator (conectori DB-25).

2.8.2. Cabluri adaptoare

În cazul în care cei doi conectori ai unei legături seriale sunt de tipuri diferite, de exemplu, DB-25 la un capăt şi DB-9 la celălalt capăt, este necesar un cablu adaptor. Deşi la calculatoarele IBM PC actuale se utilizează numai conectori DB-9, la multe periferice, cum sunt modemuri, imprimante sau plottere, se utilizează conectori DB-25. Adaptarea poate fi realizată cu o cuplă având doi conectori de tipuri diferite, sau cu un cablu adaptor având conectori de tipuri diferite la cele două capete.

Tabelul 3.3 prezintă conexiunile necesare pentru un cablu adaptor de la un conector DB-25 la un conector DB-9.

Tabelul 3.3. Conexiunile pentru un cablu adaptor de la un conector DB-25 la un conector DB-9.

Pin DB-25	Pin DB-9	Semnal	Semnificaţie
2	3	TD	Transmit Data
3	2	RD	Receive Data
4	7	RTS	Request To Send
5	8	CTS	Clear To Send
6	6	DSR	Data Set Ready
7	5	SG	Signal Ground
8	1	CD	Carrier Detect
20	4	DTR	Data Terminal Ready
22	9	RI	Ring Indicator

2.8.3. Cabluri inversoare

Cablurile inversoare trebuie utilizate pentru conectarea a două echipamente terminale de date (ETD), cum sunt două calculatoare. Aceste cabluri trebuie utilizate şi la conectarea unor tipuri de periferice la calculator, cum sunt imprimantele şi plotterele seriale, deoarece atunci când a fost concepută interfaţa serială, aceste periferice au fost considerate ca echipamente de tip ETD.

În continuare se prezintă două tipuri de cabluri inversoare. Primul tip se poate utiliza atunci când controlul fluxului de date se realizează prin metoda software, iar al doilea tip se poate utiliza atunci când controlul fluxului de date se realizează prin metoda hardware.

Multe din sistemele de comunicație serială nu utilizează toate semnalele pentru controlul fluxului de date, astfel încât la aceste sisteme numărul de conexiuni poate fi redus. În cazul cel mai simplu, când se utilizează metoda software pentru controlul fluxului de date, sunt necesare doar trei conexiuni, pentru datele transmise, datele recepționate şi masa electrică. De exemplu, dacă se utilizează conectori DB-25, trebuie conectați pinii 2, 3 şi 7, iar dacă se utilizează conectori DB-9, trebuie conectați pinii 2, 3 şi 5. Un asemenea cablu se numeşte cablu *null-modem*.

Un cablu *null-modem* se poate utiliza pentru conectarea directă a două terminale de date, de exemplu, a două calculatoare. Conexiunile sunt realizate astfel încât, din punctul de vedere al calculatorului, comunicația să se desfăşoare ca şi cum la celălalt capăt s-ar afla un modem, şi nu un alt calculator. Datele transmise de primul calculator trebuie recepționate de al doilea calculator, astfel încât pinul semnalului *TD* de la primul calculator este conectat cu pinul semnalului *RD* de la cel de-al doilea calculator, şi invers. Cei doi pini pentru masa electrică *SG* trebuie conectați împreună. La ambii conectori, pinul semnalului *DTR* este conectat cu pinii semnalelor *DSR* şi *CD* de la acelaşi capăt al conexiunii. Astfel, atunci când semnalul *DTR* este activat, semnalele *DSR* şi *CD* se activează şi ele. Semnalele *DSR* şi *CD* se activează deci în acelaşi mod ca şi cum la celălalt capăt al conexiunii s-ar afla un modem. În mod similar, la ambii conectori pinul semnalului *RTS* este conectat cu pinul semnalului *CTS* de la acelaşi capăt al conexiunii. Deoarece calculatoarele comunică cu aceeaşi viteză, controlul fluxului de date nu este necesar.

Figura 3.7 ilustrează conexiunile necesare pentru un cablu *null-modem*, presupunând că se utilizează comunicația asincronă şi conectori DB-9 la ambele capete.

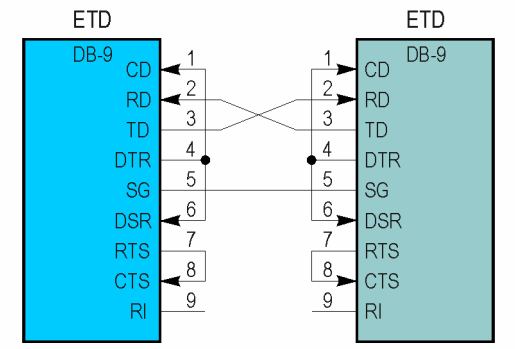

Figura 3.7. Conectarea a două echipamente terminale de date printr-un cablu *null-modem* (conectori DB-9).

Atunci când trebuie realizată o conexiune serială care utilizează metoda hardware de control al fluxului de date, cele trei linii ale cablului *null-modem* prezentat anterior nu sunt suficiente. În acest caz, trebuie să se utilizeze un cablu inversor cu conexiuni suplimentare pentru această metodă de control. Pinii de date şi pinul pentru masa electrică sunt conectați în mod similar ca şi la cablul *null-modem*. La ambii conectori, pinul semnalului *DTR* este conectat cu pinii semnalelor *DSR* şi *CD* de la celălalt capăt al conexiunii. Prin această conexiune, fiecare din cele două terminale de date de la capetele conexiunii poate determina momentul în care celălalt terminal de date este pregătit. Semnalele utilizate pentru controlul fluxului de date sunt *RTS* şi *CTS*. La ambii conectori, pinul semnalului *RTS* este conectat cu pinul semnalului *CTS* de la celălalt capăt al conexiunii. Această conexiune asigură controlul fluxului de date prin metoda hardware, în modul descris în secțiunea 2.6.1.

Figura 3.8 ilustrează legăturile necesare pentru un cablu inversor care permite un control hardware al fluxului de date, presupunând că se utilizează conectori DB-9 la ambele echipamente.

Observație

- De multe ori, cablul inversor cu conexiunile ilustrate în Figura 3.8 este numit, în mod eronat, cablu *null-modem*.

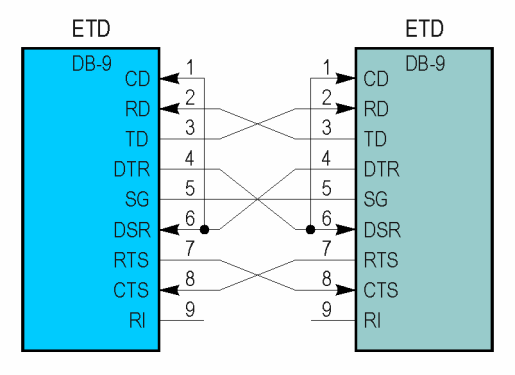

Figura 3.8. Conectarea a două echipamente terminale de date printr-un cablu inversor pentru controlul fluxului de date prin hardware (conectori DB-9).

2.9. Circuite UART

Componenta principală a unui port serial este un circuit UART (*Universal Asynchronous Receiver/Transmitter*). Un asemenea circuit realizează conversia datelor paralele de la calculator în formatul necesar pentru transmisia serială şi conversia datelor seriale recepţionate în formatul paralel utilizat de calculator. Circuitul adaugă biţii de START, de STOP şi bitul de paritate la datele seriale transmise şi detectează aceşti biţi în cadrul datelor seriale recepţionate.

La calculatoarele IBM PC originale şi calculatoarele IBM PC/XT au fost utilizate circuite UART din familia 8250. Începând cu primele sisteme de 16 biţi, a fost utilizat circuitul UART 16450, produs de firma *National Semiconductor*. Acest circuit este compatibil cu cele din familia 8250 din punctul de vedere al registrelor, dar permite comunicaţia cu viteze mai ridicate. La calculatoarele IBM PS/2 a fost utilizat circuitul UART 16550, care a fost adoptat şi pentru sistemele bazate pe procesorul 80386 şi următoarele. Circuitul 16550 funcţiona similar ca şi circuitele 8250 şi 16450, dar conţinea în plus câte o memorie FIFO de 16 octeţi pentru transmisie şi recepţie. Această memorie permitea viteze de comunicaţie superioare faţă de cele permise de circuitele anterioare. Începând cu sistemele bazate pe procesorul Pentium, circuitul UART 16550 (sau o versiune mai recentă a acestuia) a fost inclus în setul de circuite de pe placa de bază. Versiuni îmbunătăţite ale circuitului UART 16550 sunt circuitele 16650, 16750 şi 16850.

În continuare se descriu principalele caracteristici ale unor circuite UART.

8250

Acest circuit a fost utilizat la primele calculatoare IBM PC. Circuitul nu dispunea de un buffer de transmisie şi nici de recepţie, motiv pentru care viteza de comunicaţie asigurată era redusă. Circuitul avea unele defecte de funcţionare. Sistemele ROM BIOS ale calculatoarelor IBM PC şi XT au fost realizate astfel încât să ţină cont de aceste defecte.

8250A

La această versiune, s-au corectat unele defecte ale circuitului UART 8250, printre care şi un defect de funcţionare al registrului de validare a întreruperilor. Deoarece sistemele ROM BIOS ale calculatoarelor IBM PC şi XT ţineau cont de acest defect, circuitul 8250A nu funcţiona corect cu aceste calculatoare. Circuitul 8250A funcţiona cu calculatoarele IBM PC/AT, dar nu funcţiona corespunzător începând de la viteza de 9.600 biţi/s, din cauza lipsei bufferelor de transmisie şi de recepţie.

8250B

La această ultimă versiune a circuitelor din familia 8250 s-au corectat defectele întâlnite la cele două versiuni anterioare. Defectul de funcţionare al registrului de validare a întreruperilor din versiunea originală 8250 a fost reintrodus, pentru ca circuitul să funcţioneze în modul aşteptat de

sistemele ROM BIOS ale calculatoarelor IBM PC şi XT. Nici acest circuit nu funcţiona corespunzător începând de la viteza de 9.600 biţi/s.

16450

Acest circuit a fost utilizat la primele calculatoare IBM PC/AT. Circuitul permitea viteze de comunicaţie superioare datorită unor buffere de transmisie şi de recepţie de câte 1 octet. La acest circuit, a fost adăugat la setul de registre un registru de lucru de 1 octet, ca memorie temporară.

16550

Reprezintă varianta îmbunătăţită a circuitului 16450, conţinând un buffer de transmisie şi unul de recepţie de câte 16 octeţi, sub forma unor memorii FIFO. Circuitul permite de asemenea transferuri prin canale multiple DMA. Versiunea iniţială a acestui circuit nu permitea utilizarea memoriilor FIFO, defect care a fost corectat începând cu versiunea 16550A. Ultima versiune produsă de *National Semiconductor* este 16550D. Prin utilizarea memoriilor FIFO, vitezele de comunicaţie pot fi crescute in mod semnificativ, eliminându-se pierderea unor caractere la vitezele mai ridicate. Viteza maximă de comunicaţie permisă de circuitul 16550 este de 115.200 biţi/s.

16650, 16750 şi 16850

Au fost produse diferite versiuni îmbunătăţite ale circuitului 16550, care sunt compatibile cu acesta, dar conţin memorii FIFO de dimensiuni mai mari:

- Circuitul 16650 conţine două memorii FIFO de câte 32 octeţi;
- Circuitul 16750 conţine două memorii FIFO de câte 64 octeţi;
- Circuitul 16850 conţine două memorii FIFO de câte 128 octeţi.

Aceste circuite permit viteze de comunicaţie superioare, de 230,4 Kbiţi/s (16650), 460,8 Kbiţi/s (16750), respectiv 921,6 Kbiţi/s (16850). Utilizarea acestor circuite este recomandată în cazul unei legături seriale externe de viteză ridicată, cum sunt cele realizate printr-un adaptor ISDN sau un modem de 56 Kbiţi/s.

2.10. Porturile seriale ale calculatoarelor IBM PC

Sistemul ROM BIOS al calculatoarelor IBM PC originale permitea utilizarea a două porturi seriale, cu numele COM1 şi COM2. Ulterior, numărul acestor porturi a fost extins cu încă două, cu numele COM3 şi COM4. Sistemul de operare *Windows* 3.1 permitea instalarea a până la nouă porturi seriale în calculator, acest număr fiind extins la 128 începând cu sistemul de operare *Windows* 95. Aceste porturi sunt gestionate cu ajutorul driverelor de dispozitiv care le controlează.

Accesul la porturile seriale se poate realiza prin funcţii BIOS (întreruperea 14h), prin funcţii ale sistemului de operare, sau direct prin registrele circuitelor UART. Fiecare circuit UART asociat unui port serial dispune de un număr de opt registre de I/E începând de la adresa de bază a portului serial. ROM BIOS memorează adresele de bază ale porturilor seriale COM1..COM4 în 4 cuvinte succesive de 16 biţi, începând cu adresa 0000:0400h pentru portul COM1.

Adresele de bază ale porturilor seriale COM1..COM4 sunt indicate în Tabelul 3.4. În general, adresele porturilor COM1 şi COM2 sunt fixe, având valorile indicate în acest tabel. Adresele porturilor COM3 şi COM4 pot fi diferite uneori de cele indicate. În Tabelul 3.4 se indică şi nivelele de întrerupere utilizate de porturile seriale COM1..COM4.

Tabelul 3.4. Asignarea standard a adreselor de bază şi a nivelelor de întrerupere la porturile seriale.

Port serial	Adresa de bază	Întrerupere
COM1	3F8h	IRQ 4
COM2	2F8h	IRQ 3
COM3	3E8h	IRQ 4
COM4	2E8h	IRQ 3

În principiu, fiecare port serial necesită propriul nivel de întrerupere. În cazul în care există mai mult de două porturi seriale în calculator, poate fi necesară partajarea unor nivele de întrerupere. De exemplu, portul COM3 partajează întreruperea de nivel 4 cu portul COM1, iar portul COM4 partajează întreruperea de nivel 3 cu portul COM2. La sistemele bazate pe magistrala ISA, nu este posibilă partajarea unui nivel de întrerupere de către mai multe echipamente, astfel încât pot apare conflicte la apariția unor întreruperi simultane. Pentru evitarea acestor conflicte, se recomandă asignarea nivelului de întrerupere IRQ 10 pentru portul COM3 şi a nivelului IRQ 11 pentru portul COM4 (dacă aceste nivele sunt disponibile).

În cazul în care este necesară adăugarea unor porturi seriale suplimentare în calculator sub forma unei plăci de extensie, pentru aceste porturi trebuie selectate nivele de întrerupere diferite de IRQ 3 şi IRQ 4. Magistrala PCI permite partajarea nivelelor de întrerupere, astfel încât la plăcile de extensie actuale bazate pe magistrala PCI este posibilă utilizarea unui singur nivel de întrerupere fără apariția unor conflicte.

2.11. Registrele circuitelor UART 16x50

Registrele circuitelor UART sunt accesibile prin instrucțiuni de I/E. Pentru primul port serial, adresele registrelor sunt cuprinse între 3F8h şi 3FFh, iar pentru al doilea port între 2F8h şi 2FFh. Primele două adrese permit accesul la mai multe registre ale circuitului. Există şi registre care nu sunt accesibile prin program.

Adresele registrelor accesibile prin program pentru primele două porturi seriale, modul de acces la acestea (R – citire, W – scriere, R/W – citire/scriere), abrevierile şi denumirile lor sunt prezentate în Tabelul 3.5. Coloana *Offset* indică deplasamentul adresei fiecărui registru față de adresa de bază a portului serial. Coloana DLAB (*Divisor Latch Access Bit*) reprezintă valoarea bitului 7 al registrului de control al liniei (LCR). Atunci când este setat la 1, acest bit permite accesul la două registre utilizate pentru setarea vitezei de comunicație.

Tabelul 3.5. Registrele circuitelor UART 16x50.

Adr. COM1	Adr. COM2	Offset	DLAB	Acces	Abreviere	Denumire
3F8h	2F8h	+ 0	0	W	THR	Transmitter Holding Register
			0	R	RBR	Receiver Buffer Register
			1	R/W	–	Divisor Latch Register LSB
3F9h	2F9h	+ 1	0	R/W	IER	Interrupt Enable Register
			1	R/W	–	Divisor Latch Register MSB
3FAh	2FAh	+ 2	X	R	IIR	Interrupt Identification Register
			X	W	FCR	FIFO Control Register
3FBh	2FBh	+ 3	X	R/W	LCR	Line Control Register
3FCh	2FCh	+ 4	X	R/W	MCR	Modem Control Register
3FDh	2FDh	+ 5	X	R	LSR	Line Status Register
3FEh	2FEh	+ 6	X	R	MSR	Modem Status Register
3FFh	2FFh	+ 7	X	R/W	–	Scratch Register

Observație

- Circuitele 16650, 16750 şi 16850 conțin registre suplimentare față de cele indicate în Tabelul 3.5. Organizarea acestor registre poate varia de la un producător la altul, motiv pentru care aceste registre nu sunt descrise în acest document.

THR - Transmitter Holding Register

Reprezintă bufferul de transmisie, fiind selectat dacă bitul DLAB al registrului LCR este 0. Caracterul care trebuie transmis va fi înscris în acest registru. Dacă registrul TSR (*Transmitter Shift Register*) se goleşte (deci, circuitul poate începe transmisia unui nou caracter), conținutul registrului

THR (sau, dacă memoriile FIFO sunt validate, un octet din memoria FIFO de transmisie) este transferat în registrul TSR şi se începe transmisia caracterului pe linie.

Dacă registrul THR se goleşte (deci, poate fi înscris cu un nou caracter), circuitul generează o întrerupere dacă generarea acestui tip de întrerupere este validată. Starea acestui registru se poate determina prin testarea bitului 5 al registrului LSR.

TSR - Transmitter Shift Register

Este un registru intern care nu este accesibil prin program. La terminarea transmisiei unui caracter, conținutul registrului THR (sau, dacă memoriile FIFO sunt validate, un octet din memoria FIFO de transmisie) este transferat în mod automat în registrul TSR şi se începe transmisia acestuia.

Dacă registrul TSR se goleşte (deci, poate începe transmisia unui nou caracter), dar registrul THR este gol (sau, dacă memoriile FIFO sunt validate şi memoria FIFO de transmisie s-a golit sub un nivel de prag), circuitul generează o întrerupere în cazul în care acest tip de întrerupere este validată. Starea acestui registru se poate determina prin testarea bitului 6 al registrului LSR.

RBR - Receiver Buffer Register

Reprezintă bufferul de recepție, fiind selectat dacă bitul DLAB al registrului LCR este 0. Un caracter recepționat este depus în acest registru. Existența unui caracter în registrul RBR se poate determina prin testarea bitului 0 al registrului LSR. Dacă memoriile FIFO nu sunt validate, caracterul recepționat trebuie preluat de programul de recepție din registrul RBR înaintea terminării recepției unui nou caracter. În caz contrar, va apare o eroare de suprapunere (*overrun error*). Recepția unui caracter determină generarea unei întreruperi, dacă acest tip de întrerupere este validată. Dacă memoriile FIFO sunt validate, este posibil ca întreruperea de recepție să fie generată doar după recepționarea unui anumit număr de caractere în memoria FIFO de recepție. Această valoare de prag poate fi programată prin setarea biților 7..6 ai registrului FCR.

RSR - Receiver Shift Register

Este un registru intern care nu este accesibil prin program. Fiecare caracter este recepționat în acest registru. La terminarea recepției caracterului, conținutul registrului RSR este transferat în mod automat în registrul de recepție RBR dacă memoriile FIFO nu sunt validate. Dacă aceste memorii sunt validate, caracterele recepționate sunt depuse în memoria FIFO de recepție.

Divisor Latch Register LSB & MSB

Conțin valoarea cu care trebuie divizată frecvența ceasului propriu al circuitului 16x50 (1,8432 MHz) pentru a se obține debitul binar necesar. Registrul LSB conține octetul mai puțin semnificativ al divizorului, iar registrul MSB conține octetul mai semnificativ. Cele două registre divizor sunt accesibile dacă bitul DLAB al registrului LCR este 1. Pentru calculul divizorului trebuie să se țină cont de factorul de ceas al circuitului 16x50, de obicei 16 (debitul binar este de 16 ori mai mic decât frecvența obținută prin divizare). Se poate utiliza următoarea formulă:

$$Divizor = 1.843.200 / (DebitBinar * 16)$$

Tabelul 3.6 conține divizorii corespunzători diferitelor debite binare.

Tabelul 3.6. Divizorii frecvenței de 1,8432 MHz pentru diferite debite binare.

Debit binar (biți/s)	Divizor	Debit binar (biți/s)	Divizor
50	0900h	4.800	0018h
150	0300h	9.600	000Ch
300	0180h	19.200	0006h
600	00C0h	38.400	0003h
1.200	0060h	57.600	0002h
2.400	0030h	115.200	0001h

IER - Interrupt Enable Register

Reprezintă registrul de validare a întreruperilor, fiind selectat dacă bitul DLAB al registrului LCR este 0. Circuitul poate genera cinci tipuri de întreruperi, cu nivele de prioritate diferite. Registrul IER permite validarea independentă a generării acestor întreruperi. Structura acestui registru este ilustrată în Figura 3.9.

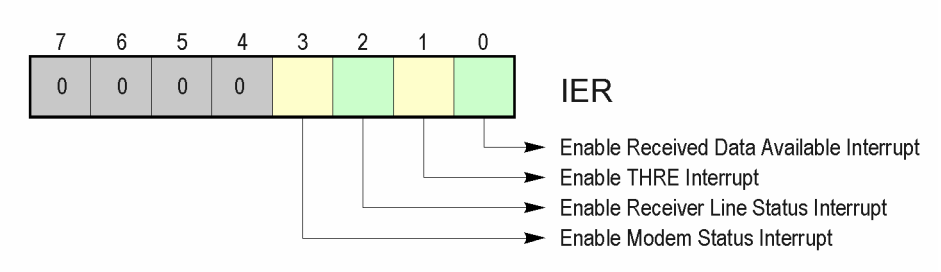

Figura 3.9. Registrul de validare a întreruperilor IER.

Biții registrului IER sunt descrişi în continuare.

- Biții 7..4 sunt neutilizați (0).
- Bitul 3 (Enable Modem Status Interrupt) validează prin valoarea 1 generarea unei întreruperi la modificarea registrului de stare al modemului (MSR).
- Bitul 2 (Enable Receiver Line Status Interrupt) validează prin valoarea 1 generarea unei întreruperi la modificarea registrului de stare a liniei (LSR), de obicei, la apariția unor erori de recepție.
- Bitul 1 (Enable Transmitter Holding Register Empty Interrupt) validează prin valoarea 1 generarea întreruperii la golirea registrului THR, deci, după transferarea conținutului acestui registru în registrul TSR. La apariția acestei întreruperi se poate înscrie un nou caracter în registrul THR.
- Bitul 0 (Enable Received Data Available Interrupt) validează prin valoarea 1 generarea întreruperii la recepția unui caracter. Dacă memoriile FIFO sunt validate, acest bit validează şi generarea întreruperii de depăşire a timpului (*time-out*). Această întrerupere este descrisă în secțiunea următoare.

IIR - Interrupt Identification Register

Reprezintă registrul de identificare a întreruperilor. Acest registru poate fi accesat doar pentru citire. Deşi întreruperile generate de circuitul 16x50 apar pe un singur nivel de întrerupere (IRQ 4 pentru primul port serial şi IRQ 3 pentru portul al doilea), aceste întreruperi pot avea patru tipuri de cauze, cu nivele de prioritate diferite. Identificarea cauzei întreruperii se poate realiza prin testarea biților 3..0 din registrul de identificare a întreruperilor (IIR). Ceilalți biți ai registrului indică starea memoriilor FIFO ale circuitului UART.

Semnificația biților registrului IIR care permit identificarea cauzei unei întreruperi este indicată în Tabelul 3.7.

Din Tabelul 3.7 se observă că circuitul se poate utiliza şi prin interogare, testând periodic bitul 0 al registrului de identificare a întreruperilor. Dacă acest bit este 0, înseamnă că a apărut o întrerupere.

Nivelul de prag al memoriei FIFO de recepție se referă la numărul de caractere care trebuie recepționate înainte ca circuitul să genereze o întrerupere de recepție, dacă acest tip de întrerupere este validată. Setarea acestui nivel de prag este descrisă în secțiunea dedicată registrului de control al memoriilor FIFO (FCR).

Tabelul 3.7. Biții registrului IIR utilizați pentru identificarea cauzei unei întreruperi.

Biți 3..0	Prioritate	Tip întrerupere	Cauza întreruperii	Resetarea întreruperii
0 0 0 1	–	–	Nu există întrerupere	–
0 0 0 0	3 (minimă)	Modificare stare modem	Modificare CTS sau DSR sau RI sau CD	Citire registru de stare modem (MSR)
0 0 1 0	2	Terminare transmisie caracter	Registrul THR este gol	Citire IIR sau scriere caracter în THR
0 1 0 0	1	Recepție caracter	Registrul RBR conține un caracter recepționat sau memoria FIFO de recepție s-a umplut peste nivelul de prag	Citire registru RBR sau golire memorie FIFO de recepție sub nivelul de prag
1 1 0 0	1	Depăşirea timpului (time-out)	Nu s-au citit sau depus caractere din/în memoria FIFO de recepție pe durata a 4 caractere şi există cel puțin un caracter în memoria FIFO	Citire registru RBR
0 1 1 0	0 (maximă)	Modificare stare linie	Eroare de suprapunere, de încadrare sau de paritate, sau transmisie spații (Break)	Citire registru de stare linie (LSR)

La o operație de recepție cu memoriile FIFO validate, circuitul UART va genera o întrerupere chiar dacă memoria FIFO de recepție conține un număr de caractere mai mic decât valoarea de prag, dar nu s-au mai recepționat caractere într-un timp egal cu durata necesară transferului a 4 caractere. Aceasta reprezintă întreruperea de depăşire a timpului (*time-out*) şi a fost prevăzută pentru cazul în care transmițătorul opreşte transmisia caracterelor pentru a aştepta un răspuns de la receptor. Fără această întrerupere, ar fi posibil ca receptorul să nu sesizeze recepția caracterelor, deoarece memoria FIFO de recepție nu conține un număr suficient de caractere pentru a se genera o întrerupere de recepție. Bitul din registrul IIR care indică această cauză de întrerupere poate fi neglijat, deoarece circuitul UART va indica şi faptul că sunt disponibile caractere în bufferul de recepție.

Biții 7..4 ai registrului IIR sunt descrişi în continuare.

- Biții 7..6 indică starea memoriilor FIFO:

  00: Nu există memorii FIFO (circuitul este de tip 8250 sau 16450).
  01: Combinație rezervată.
  10: Memoriile FIFO sunt validate, dar nu sunt utilizabile. Această situație poate apare la circuitul 16550, din cauza defectului de utilizare a memoriilor FIFO la acest circuit.
  11: Memoriile FIFO sunt validate şi operaționale.

- Biții 5..4 sunt rezervați (0).

FCR - FIFO Control Register

Reprezintă registrul de control al memoriilor FIFO, fiind prezent la circuitul 16550 şi la următoarele. Structura registrului FCR este ilustrată în Figura 3.10.

Biții registrului FCR sunt descrişi în continuare.

- Biții 7..6 (Receiver Trigger) permit setarea numărului de caractere recepționate în memoria FIFO de recepție după care circuitul va genera o întrerupere de recepție, dacă acest tip de întrerupere este validată. Dacă acest număr este setat la o valoare mai mare decât 1, circuitul nu va genera o întrerupere după fiecare caracter recepționat, ceea ce va reduce timpul necesar tratării întreruperilor. Semnificația biților 7..6 este următoarea:

  00: Întreruperea se generează după fiecare caracter recepționat;
  01: Întreruperea se generează după 4 caractere recepționate;
  10: Întreruperea se generează după 8 caractere recepționate;
  11: Întreruperea se generează după 14 caractere recepționate.

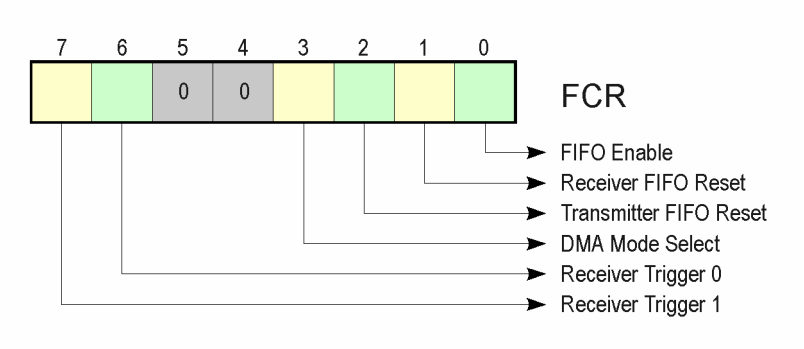

Figura 3.10. Registrul de control al memoriilor FIFO FCR.

- Biţii 5..4 sunt rezervaţi (0).
- Bitul 3 (DMA Mode Select), prezent la circuitul 16550 şi următoarele, permite selectarea modului de transfer prin DMA. Atunci când memoriile FIFO sunt validate, există două moduri de transfer prin DMA care pot fi selectate, modul 0 şi modul 1. Modul 0, selectat dacă bitul 3 al registrului FCR este 0, permite transferuri DMA singulare, de câte un singur cuvânt. Modul 1, selectat dacă bitul 3 al registrului FCR este 1, permite transferuri DMA multiple, acestea fiind executate în mod continuu până când memoria FIFO de recepţie se goleşte sau memoria FIFO de transmisie se umple. La circuitul 16450, este permis doar modul 0.
- Bitul 2 (Transmitter FIFO Reset) permite ştergerea memoriei FIFO de transmisie prin setarea acestui bit la 1.
- Bitul 1 (Receiver FIFO Reset) permite ştergerea memoriei FIFO de recepţie prin setarea acestui bit la 1.
- Bitul 0 (FIFO Enable) validează prin valoarea 1 memoriile FIFO de transmisie şi de recepţie. În mod implicit, aceste memorii sunt invalidate, pentru compatibilitate cu circuitele UART 8250 şi 16450. Prin setarea acestui bit la 0 se vor invalida memoriile FIFO, datele memorate în acestea fiind pierdute.

Observaţie

- În cazul în care nu se utilizează memoriile FIFO, sau biţii 7..6 ai registrului FCR sunt setaţi astfel încât să se genereze o întrerupere după fiecare caracter recepţionat, în rutina de tratare întreruperii de recepţie este suficient să se citească registrul de recepţie RBR o singură dată. Dacă se utilizează memoriile FIFO, în rutina de tratare a întreruperii de recepţie trebuie implementată o buclă de program pentru citirea câte unui caracter din registrul de recepţie RBR, cât timp există un caracter în acest registru. Existenţa unui caracter în registrul de recepţie RBR este indicată prin faptul că bitul 0 din registrul LSR este setat. După citirea unui caracter din registrul RBR, circuitul va depune în mod automat următorul caracter din memoria FIFO, dacă mai există caractere în această memorie.

LCR - Line Control Register

Prin înscrierea registrului de control al liniei se pot stabili parametrii comunicaţiei seriale. Structura registrului LCR este ilustrată în Figura 3.11.

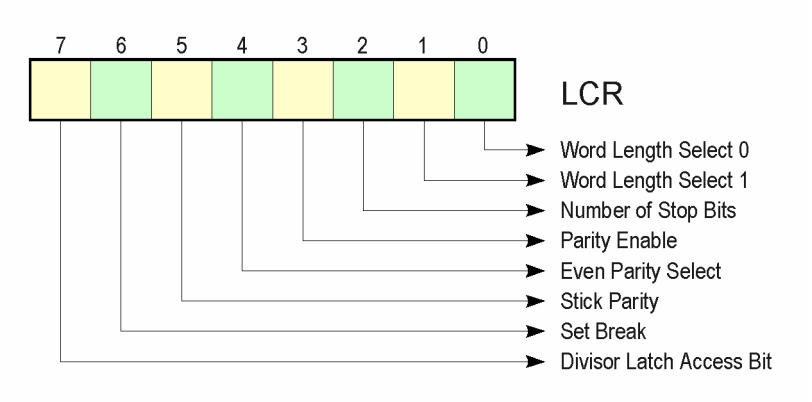

Figura 3.11. Registrul de control al liniei LCR.

Semnificația biților registrului LCR este următoarea:

- Bitul 7 (Divisor Latch Access Bit) modifică rolul registrelor accesibile prin adresele 3F8h (2F8h) şi 3F9h (2F9h). Dacă acest bit este 0, aceste registre sunt *Transmitter / Receiver Buffer Register*, respectiv *Interrupt Enable Register*, iar dacă bitul este 1, registrele sunt cele utilizate pentru înscrierea divizorului care stabileşte viteza de transmisie.
- Bitul 6 (Set Break). Dacă este 1, circuitul forțează linia de transmisie la nivelul 0 logic (spațiu). Aceasta corespunde stării "*break*" a liniei, care permite atenționarea unui terminal aflat la distanță printr-o întrerupere generată la detectarea acestei stări a liniei. Linia poate fi readusă în starea normală prin resetarea bitului 6.
- Bitul 5 (Stick Parity) permite transmiterea sau aşteptarea unor biți de paritate cu valoare fixă, 0 sau 1:

  0: Verificarea obişnuită a parității, conform biților Parity Enable şi Even Parity Select;
  1: Dacă bitul Parity Enable este 1, se transmit sau se verifică biți cu valoare fixă în locul bitului de paritate, conform bitului Even Parity Select. Dacă bitul Even Parity Select este 0, bitul de paritate este întotdeauna 1, iar dacă bitul Even Parity Select este 1, bitul de paritate este întotdeauna 0.
- Bitul 4 (Even Parity Select) indică tipul parității utilizate, dacă generarea şi verificarea bitului de paritate este validată prin bitul Parity Enable:

  0: Paritate impară;
  1: Paritate pară.
- Bitul 3 (Parity Enable) validează sau invalidează generarea şi verificarea bitului de paritate:

  0: Generarea şi verificarea bitului de paritate este invalidată;
  1: Generarea şi verificarea bitului de paritate este validată.
- Bitul 2 (Number of Stop Bits) indică numărul biților de STOP generați de circuitul UART la transmisie sau aşteptați de circuit la recepție:

  0: 1 bit de STOP;
  1: 2 biți de STOP (1,5 biți dacă lungimea caracterelor este de 5 biți).

  Receptorul testează doar primul bit de STOP, indiferent de numărul biților de STOP selectați.
- Biții 1..0 (Word Length Select) specifică lungimea caracterelor transmise sau recepționate:

  00: 5 biți/caracter;
  01: 6 biți/caracter;
  10: 7 biți/caracter;
  11: 8 biți/caracter.

MCR - Modem Control Register

Registrul de control al modemului MCR (Figura 3.12) se utilizează pentru comanda comunicației cu modemul.

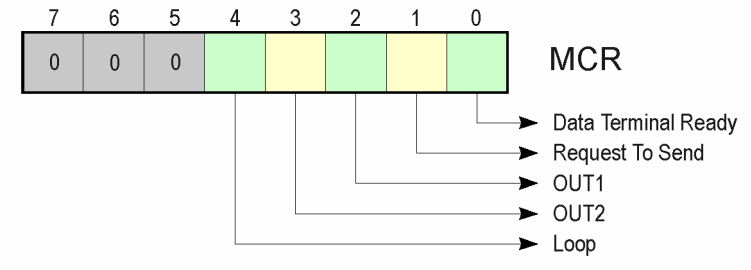

Figura 3.12. Registrul de control al modemului MCR.

Semnificația biților registrului MCR este următoarea:

- Biții 7..5 sunt neutilizați (0).
- Bitul 4 (Loop) permite testarea circuitului UART şi a programelor de comunicație. Prin setarea acestui bit la 1 se vor efectua următoarele operații:
  1. Ieşirea serială a transmițătorului este plasată în starea 1 logic.
  2. Intrarea serială a receptorului este deconectată.
  3. Datele de la ieşirea registrului TSR vor fi recepționate în bufferul de recepție RBR.
  4. Liniile de intrare pentru controlul modemului *DSR*, *CTS*, *RI* şi *DCD* sunt deconectate, iar comanda lor aparentă se poate realiza cu biții 0..3 ai registrului de control al modemului (Data Terminal Ready, Request To Send, OUT1, respectiv OUT2). Dacă circuitul este programat astfel încât întreruperile să fie validate, modificarea acestor biți va determina generarea întreruperilor ca şi în cazul în care semnalele ar fi activate de modem.
- Biții 2 şi 3 (OUT1 şi OUT2) se pot utiliza pentru implementarea unei comunicații definite de utilizator.
- Bitul 1 (Request To Send) activează prin valoarea 1 semnalul *RTS* al interfeței.
- Bitul 0 (Data Terminal Ready) activează prin valoarea 1 semnalul *DTR* al interfeței.

Observație

- Semnalele *DTR*, *RTS*, *OUT1* şi *OUT2* sunt active în starea 0 logic.

LSR - Line Status Register

Acest registru indică starea liniei de comunicație. Biții 6..5 se referă la transmisie, iar biții 4..0 se referă la recepție (Figura 3.13). Semnificația biților registrului LSR este următoarea:

- Bitul 7 (Error in Receiver FIFO) este setat la 1 atunci când memoriile FIFO sunt validate şi a apărut cel puțin o eroare de paritate, eroare de încadrare sau o condiție de "*break*" la recepția caracterelor aflate în memoria FIFO de recepție.
- Bitul 6 (Transmitter Shift Register Empty) este setat la 1 dacă atât registrul THR cât şi registrul TSR s-au golit. Dacă memoriile FIFO sunt validate, acest bit este setat la 1 atunci când atât memoria FIFO de transmisie cât şi registrul TSR s-au golit.
- Bitul 5 (Transmitter Holding Register Empty) este setat la 1 atunci când conținutul registrului THR este depus în registrul TSR şi se începe transmisia caracterului. Aceasta indică faptul că circuitul UART este pregătit pentru a accepta un nou caracter pentru transmisie. La golirea

registrului THR, circuitul UART generează o întrerupere dacă acest tip de întrerupere este validată. Bitul este resetat la înscrierea unui caracter în registrul THR. Dacă memoriile FIFO sunt validate, acest bit este setat la 1 atunci când memoria FIFO de transmisie se goleşte şi este resetat la 0 atunci când se înscrie cel puţin un caracter în memoria FIFO de transmisie.

- Bitul 4 (Break Interrupt) este setat la 1 dacă sunt sesizate spaţii (0 logic) pe linie pentru o perioadă mai mare decât cea necesară pentru transmisia unui caracter. Se depune un octet cu valoarea 0 în bufferul de recepţie şi se generează o întrerupere. Bitul este resetat prin citirea registrului LSR.
- Bitul 3 (Framing Error) indică o eroare de încadrare, fiind setat la 1 dacă se recepţionează un caracter fără biţii de STOP corespunzători. La recepţie se testează numai primul bit de STOP, indiferent de numărul biţilor de STOP programaţi. La detectarea acestei erori, circuitul încearcă să se resincronizeze. Bitul este resetat prin citirea registrului LSR.
- Bitul 2 (Parity Error) indică o eroare de paritate, fiind setat la 1 dacă se recepţionează un caracter cu paritatea diferită de cea aşteptată. Bitul este resetat prin citirea registrului LSR.
- Bitul 1 (Overrun Error) indică o eroare de suprapunere. Este setat la 1 dacă se recepţionează un caracter înaintea citirii caracterului din registrul RBR de către unitatea centrală. În acest caz, se pierd unul sau mai multe caractere. Eroarea de suprapunere, ca şi celelalte erori, generează o întrerupere. Bitul este resetat în urma citirii registrului LSR. Dacă memoriile FIFO sunt validate şi memoria FIFO de recepţie se umple peste nivelul de prag, o eroare de suprapunere va fi semnalată doar după ce memoria FIFO s-a umplut şi următorul caracter a fost recepţionat în registrul RSR.
- Bitul 0 (Data Ready) este setat la 1 atunci când a fost recepţionat un caracter şi acesta a fost transferat în registrul RBR sau în memoria FIFO. Bitul este resetat automat în urma citirii caracterului din registrul RBR sau în urma citirii tuturor caracterelor din memoria FIFO de recepţie. Recepţia unui caracter va genera o întrerupere dacă acest tip de întrerupere este validată.

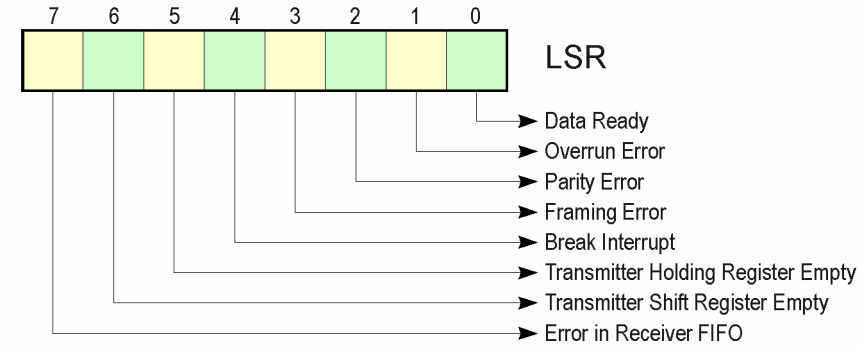

Figura 3.13. Registrul de stare a liniei LSR.

MSR - Modem Status Register

Acest registru conţine informaţii despre starea modemului (Figura 3.14). Semnificaţia biţilor registrului MSR este următoarea:

- Biţii 7..4 indică starea curentă a semnalelor *CD*, *RI*, *DSR*, respectiv *CTS*. Un semnal activ este indicat prin setarea la 1 a bitului corespunzător din registrul MSR. Dacă bitul Loop din registrul MCR este 1, starea biţilor 7, 6, 5, 4 din registrul MSR este echivalentă cu starea biţilor OUT2, OUT1, Data Terminal Ready, respectiv Request To Send din registrul MCR.
- Biţii 3..0 indică modificarea stării semnalelor *CD*, *RI*, *DSR*, respectiv *CTS*, de la ultima citire a registrului MSR. Aceşti biţi sunt resetaţi la citirea registrului MSR.

Observație

- Semnalele *CTS*, *DSR*, *RI* şi *CD* sunt active în starea 0 logic.

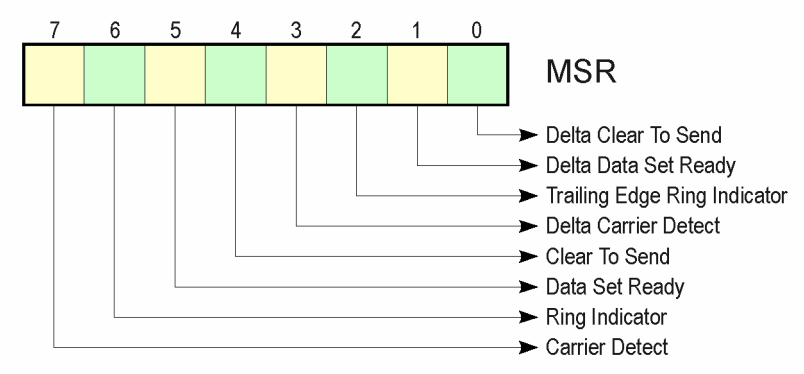

Figura 3.14. Registrul de stare a modemului MSR.

Scratch Register

Acest registru nu este utilizat pentru comunicație. Poate fi utilizat pentru memorarea temporară a unui octet.

3. Desfăşurarea lucrării

3.1. Răspundeți la următoarele întrebări:

a. Care este deosebirea dintre unitatea de măsură a vitezei de modulație şi cea a vitezei de comunicație ?

b. Cum se poate realiza sincronizarea între ceasul receptorului şi cel al transmițătorului în cazul comunicației sincrone ?

c. Cum se poate asigura corectitudinea blocurilor de date transmise în cazul unei comunicații sincrone ?

d. Care este funcția fiecăruia din următoarele semnale ale interfeței seriale: *DTR*, *DSR*, *RTS* şi *CTS* ?

e. Descrieți protocolul utilizat pentru controlul fluxului de date prin metoda hardware în cazul unei legături seriale directe între două echipamente (fără modemuri şi linie telefonică).

3.2. Scrieți o funcție pentru transmisia unui caracter prin portul serial COM1. Utilizați această funcție pentru scrierea unui program care transmite un şir de caractere prin portul serial COM1. Conectați un cablu serial între porturile COM1 ale două calculatoare (sau între porturile COM1 şi COM2 ale aceluiaşi calculator). Verificați parametrii setați de sistemul de operare pentru portul serial al calculatorului care va fi utilizat ca transmițător. Lansați în execuție programul *HyperTerminal* pe calculatorul care va fi utilizat ca receptor şi creați o conexiune cu aceiaşi parametri ca şi cei ai portului serial al calculatorului transmițător. Lansați apoi în execuție programul de transmisie şi verificați funcționarea acestuia urmărind caracterele afişate în fereastra programului *HyperTerminal*.

3.3. Scrieți o funcție pentru recepția unui caracter prin portul serial COM1. Utilizați această funcție pentru scrierea unui program care recepționează un şir de caractere prin portul serial COM1. Programul va afişa pe ecran fiecare linie recepționată terminată cu caracterul CR (0Dh) şi se va termina la recepția caracterului ESC (1Bh). Conectați un cablu serial între porturile COM1 ale două calculatoare (sau între porturile COM1 şi COM2 ale aceluiaşi calculator). Verificați parametrii setați de sistemul de operare pentru portul serial al calculatorului care va fi utilizat ca receptor. Lansați în execuție programul *HyperTerminal* pe calculatorul care va fi utilizat ca transmițător şi creați o conexiune cu aceiaşi parametri ca şi cei ai portului serial al calculatorului receptor. Lansați apoi în

execuție programul de recepție şi verificați funcționarea acestuia introducând linii de text în fereastra programului *HyperTerminal*.

3.4. Modificați programul de recepție scris pentru aplicația 3.3 astfel încât acesta să transmită în ecou fiecare caracter recepționat la portul serial COM1. Verificați funcționarea programului într-un mod similar cu procedura descrisă la aplicația 3.3.

3.5. Conectați două calculatoare printr-un cablu serial. Utilizați programul de transmisie scris pentru aplicația 3.2 şi programul de recepție scris pentru aplicația 3.3 pentru transmiterea unui şir de caractere de la unul din calculatoare la celălalt. Calculatorul receptor va afişa pe ecran şirul de caractere recepționat.

3.6. Scrieți o funcție pentru inițializarea portului serial COM1 cu următorii parametri: viteza de 19.200 biți/s, lungimea caracterelor de 8 biți, fără bit de paritate, 1 bit de STOP.

Bibliografie

[1] Baruch, Z., *Sisteme de intrare/ieşire, Îndrumător de lucrări de laborator*, Editura U.T.PRES, Cluj-Napoca, 1998.

[2] National Semiconductor Corp., "PC16550D Universal Asynchronous Receiver/Transmitter with FIFOs", 1995, www.national.com/ds.cgi/PC/PC16550D.pdf.

[3] Peacock, C., "Interfacing the Serial / RS232 Port", Beyond Logic, 2005, http://beyondlogic.org/serial/serial.htm.

[4] Rosch, W. L., *Hardware Bible*, Sixth Edition, Que Publishing, 2003.

[5] Strangio, C. E., "The RS232 Standard", CAMI Research Inc., Lexington, Massachusetts, 2003-2005, http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html.

[6] *** "RS-232", Wikipedia, The Free Encyclopedia, 2005, http://en.wikipedia.org/wiki/V.24.