

3. PORTUL SERIAL

Această lucrare prezintă principiile comunicației seriale și structura porturilor seriale ale calculatoarelor IBM PC, în particular a familiei 16x50 de controlere seriale utilizate pentru aceste calculatoare.

3.1. Modelul comunicației seriale

Un exemplu de sistem pentru comunicația serială este ilustrat în figura 3.1.

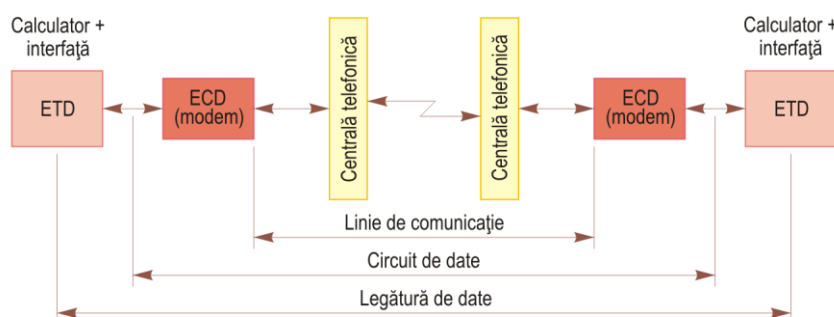


Figura 3.1. Sistem de comunicație serială.

Componentele unui sistem de comunicație serială sunt următoarele:

1. ETD – *Echipamente terminale de date* (calculatoare, terminale de date). Acestea conțin și interfețele seriale sau controlerele de comunicație.
2. ECD – *Echipamente pentru comunicația de date*. Aceste echipamente se numesc modemuri și permit calculatorului să transmită informații printr-o linie telefonică analogică. Funcțiile principale realizate de un modem sunt următoarele:
 - Conversia digital/analogică a informațiilor din calculator și conversia analog/digitală a semnalelor de pe linia telefonică analogică.
 - Modularea/demodularea unui semnal purtător. La transmisie, modemul suprapune (modulează) semnalele digitale ale calculatorului peste semnalul purtător al liniei telefonice. La recepție, modemul extrage (demodulează) informațiile transportate de semnalul purtător și le transferă calculatorului.
3. *Linia de comunicație* reprezintă o linie fizică sau o linie telefonică. Linia telefonică poate fi o linie comutată (conectată la o centrală telefonică) sau o linie închiriată (dedicată).
4. *Circuitul de date* cuprinde porțiunea dintre două echipamente terminale de date, deci, modemurile și linia de comunicație. Pe distanțe reduse, este posibilă comunicația 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 și interfețele seriale ale echipamentelor terminale de date.

În funcție de numărul de echipamente interconectate, o legătură serială poate fi *punct la punct* (două echipamente) sau *multi-punct* (mai mult de două echipamente).

3.2. Parametrii comunicației seriale

Viteza de comunicație (numită și *debit binar*) este măsurată în biți/s (bps):

$$D = \frac{1}{T} \text{ [biți/s]}$$

unde T este perioada de timp necesară pentru transmisia sau recepția unui singur bit.

Modemul reprezintă semnalele de date prin diferite stări electrice, în funcție de tipul de modulație pe care îl utilizează: frecvență, amplitudine, sau fază. Fiecare stare electrică este menținută la ieșirea modemului pentru un interval de timp numit *perioadă de modulație* (Δ). *Viteza de modulație* este inversul perioadei 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 și telegrafistului francez Jean-Maurice Baudot. Relația dintre viteza de comunicație D și 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 cazul particular când există doar două stări electrice distincte ale modemului, viteza de comunicație este egală cu viteza de modulație. În general însă, există un număr mai mare de stări electrice ale modemului, astfel încât viteza de comunicație este un multiplu al vitezei de modulație.

Viteza de modulație este confundată adesea cu viteza de comunicație (debitul binar). Viteza de modulație (exprimată în baud) este rata cu care se modifică stările electrice ale modemului într-o secundă. De exemplu, dacă se utilizează modulația în frecvență, iar frecvența semnalului purtător se poate modifica de către modem cu o rată de 2.400 de ori pe secundă, viteza de modulație este de 2.400 baud. Primele modemi codificau un bit de 0 printr-o anumită frecvență și un bit de 1 printr-o altă frecvență. În acest caz particular, viteza de modulație are aceeași valoare cu viteza de comunicație. În general însă, modemurile codifică 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 comunicație va fi de $4 \times 2.400 = 9.600$ biți/s.

3.3. Tipuri de comunicație serială

Din punctul de vedere al *direcției de transfer*, se pot distinge următoarele tipuri de comunicație serială:

- Simplex;
- Semiduplex;
- Duplex.

În cazul comunicației *simplex*, datele sunt transferate întotdeauna în aceeași 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 acest tip de conexiune, este suficientă o singură linie de transmisie (două fire de legătură). Într-o comunicație *duplex* (numită și *duplex integral*), datele se transferă simultan în ambele direcții. Primele conexiuni duplex necesitau două linii de transmisie (patru fire de legătură), dar conexiunile ulterioare necesită o singură linie.

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

- Asincronă;
- Sincronă.

3.3.1. Comunicația asincronă

Pentru a asigura sincronizarea dintre transmițător și receptor, fiecare caracter transmis este precedat de un bit de START, cu valoarea logică 0 ("space"), și este urmat de cel puțin un bit de STOP, cu valoarea logică 1 ("mark"). Biții de START și de STOP încadrează deci fiecare caracter transmis; caracterul transmis între acești doi biți reprezintă 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. Acest mod de comunicație este numit și *start-stop*.

Sincronizarea la nivel de bit se realizează cu ajutorul semnalelor de ceas locale cu aceeași frecvență. Atunci când receptorul detectează începutul unui caracter indicat prin bitul de START, pornește un oscilator de ceas local, care permite eșantionarea corectă a biților individuali ai caracterului. Eșantionarea biților se realizează aproximativ la mijlocul intervalului corespunzător fiecărui bit.

Figura 3.2 ilustrează transmisia caracterului cu codul ASCII 0x61. 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 acest exemplu, paritatea este impară. Bitul de paritate este opțional, iar în cazul în care se adaugă la caracterul transmis, paritatea poate fi selectată pentru a fi pară sau impară. Există și posibilitatea ca bitul de paritate să fie setat la 0 sau 1, indiferent de paritatea efectivă a caracterului. În exemplul ilustrat, la sfârșitul caracterului se transmit doi biți de STOP s_1 și s_2 , după care linia rămâne în starea 1 logic un timp nedefinit. Acest timp corespunde unui interval de pauză.

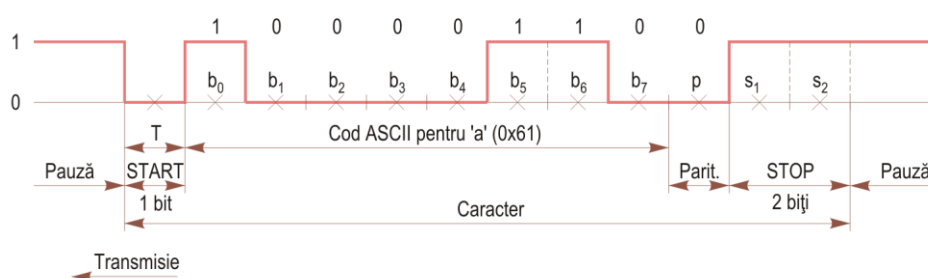


Figura 3.2. Comunicație asincronă.

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 și are dezavantajul că necesită informații suplimentare în proporție de cel puțin 25% pentru identificarea fiecărui caracter.

3.3.2. 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 și 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 recunoașterea caracterelor, din cauza lungimii blocurilor de caractere.

Pentru a se evita asemenea 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 și de la 0 la 1 în mesajul transmis. Dacă datele de transmis constau din șiruri 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, deși 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 și 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 (0x16). 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, utilizând circuite cu calare de fază PLL (*Phase-Locked Loop*)¹, pe baza tranzițiilor existente în semnalul recepționat;
- Sincronizare la nivel de caracter, asigurată prin recunoașterea anumitor caractere de sincronizare;
- Sincronizare la nivel de bloc sau mesaj, care depinde de protocolul de date utilizat.

3.4. 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–*Electronic Industries Association*). Standardul a fost elaborat pentru comunicația digitală între un calculator și 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. De exemplu, în anul 1987 a fost elaborată o nouă revizie a standardului, numită EIA RS-232D. În anul 1991, EIA și Asociația Industriei de Telecomunicații (TIA–*Telecommunications Industry Association*) au elaborat revizia E a standardului (EIA/TIA RS-232E). Revizia curentă este EIA RS-232F, publicată în anul 1997. Totuși, 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 comitetul CCITT (*Comité Consultatif International pour Téléphonie et Télégraphie*). Denumirea acestui comitet 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 și conectorii interfeței seriale.

Standardul RS-232C definește atât o comunicație asincronă, cât și una sincronă. Nu sunt definite detalii cum sunt codificarea caracterelor (ASCII, Baudot, EBCDIC), încadrarea caracterelor (lungimea caracterului, numărul biților de stop, paritatea) și nici vitezele de comunicație, deși 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; 115.200 biți/s.

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

Spre deosebire de alte tipuri de comunicație serială care sunt diferențiale², comunicația RS-232C este una obișnuită, utilizând câte un fir pentru fiecare semnal. Deși astfel se simplifică circuitele necesare interfeței, în același timp se reduce și distanța maximă de comunicație în cazul

¹ Un circuit PLL reprezintă un sistem în buclă închisă pentru controlul frecvenței unui oscilator. Funcționarea sa se bazează pe detectarea diferenței de fază între semnalele de intrare și de ieșire ale oscilatorului controlat.

² O comunicație diferențială utilizează câte o pereche de fire pentru fiecare semnal. Exemple de interfețe care utilizează comunicația diferențială sunt interfețele RS-422 și RS-485, sau magistralele USB și IEEE 1394.

unei legături directe, fără utilizarea modemurilor. Standardul RS-232C specifică o distanță maximă de 15 m. Distanța poate fi mărită 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 și +25 V;
- Valoarea logică 1 corespunde unei tensiuni negative între -3 V și -25 V.

3.5. Semnalele interfeței seriale

Interfața serială utilizează atât semnale pentru transmisia și recepția datelor, cât și semnale pentru controlul fluxului de date între un echipament terminal de date și un echipament pentru comunicația de date. Figura 3.3 prezintă semnalele interfeței seriale conform standardului RS-232C. Legătura ilustrată este cea pentru care a fost conceput inițial portul serial, și anume, conectarea unui modem la calculator. Figura indică numărul pinilor conectorilor DB-25 utilizați în cazul unei asemenea legături prin modemuri; numerele din paranteze reprezintă semnalele conform standardului V.24.

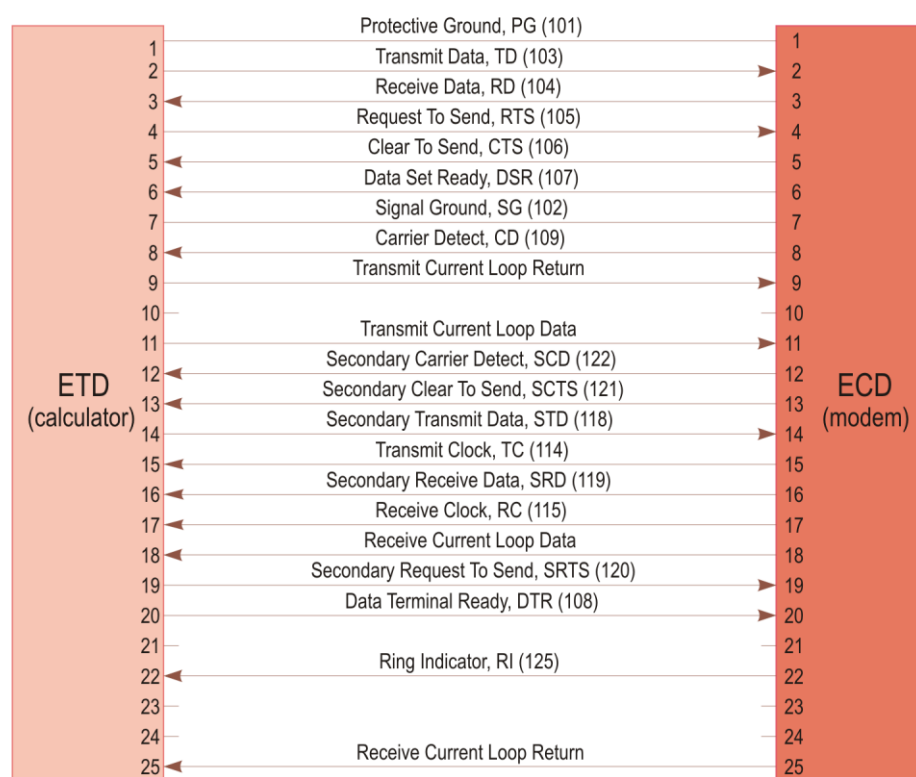


Figura 3.3. Semnalele interfeței seriale.

Cele mai importante 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 transmisia datelor este necesar ca semnalele *RTS*, *CTS*, *DTR* și *DSR* să fie active. Aceste semnale sunt activate în cadrul unei secvențe 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 și pregătit pentru comunicația de date, activează semnalul *DTR*. Modemul va răspunde la semnalul *DTR* prin activarea semnalului *DSR*.

Data Set Ready, DSR (*Modem operațional*)

Atunci când modemul sau echipamentul extern este operațional și pregătit pentru comunicația de date, activează semnalul *DSR*. Acest semnal 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, activarea semnalului *CD* indică faptul că este posibilă comunicația cu un echipament de la celălalt capăt al liniei. 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 apropiate între ele, fără utilizarea unor modemuri. Metoda utilizată se numește transmisie prin *bucă 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. Deși 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.

Secvența de stabilire a legăturii între calculator și modem constă din următoarele operații: programul activează semnalul *DTR* și așteaptă răspunsul de la modem, reprezentat de activarea semnalului *DSR*. În continuare, programul activează semnalul *RTS* și așteaptă răspunsul de la modem, reprezentat de activarea semnalului *CTS*.

3.6. Controlul fluxului de date

Pentru a fi posibilă comunicația între dispozitive cu viteze diferite, proiectanții interfeței seriale au prevăzut semnale speciale pentru controlul fluxului de date. Aceste semnale permit unui echipament oprirea și apoi reluarea transmiterii datelor la cererea echipamentului de la celălalt capăt al liniei de comunicație serială. Pe lângă această *metodă hardware* pentru controlul fluxului de date, există și o *metodă software*, bazată pe transmiterea unor caractere speciale între cele două echipamente. Atunci când echipamentul receptor (de exemplu, o imprimantă) nu mai poate primi date deoarece bufferul acestuia este plin, transmite un anumit caracter de control echipamentului transmițător (de exemplu, calculatorului). Atunci când echipamentul receptor poate primi noi date, transmite un alt caracter de control care semnalează echipamentului transmițător că poate relua transmiterea datelor.

De obicei, metoda de control care va fi utilizată de calculator poate fi selectată prin intermediul driverului software al 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 printr-un comutator. Este important să se utilizeze aceeași metodă de control atât pentru calculator, cât și pentru periferic pentru a evita pierderile de date.

3.6.1. Metoda de control hardware

Metoda de control hardware presupune utilizarea unui protocol de comunicație cu ajutorul semnalelor de control ale interfeței seriale. Protocolul utilizat se bazează pe comunicația serială prin intermediul unor modemuri și a unei linii telefonice, pentru care a fost elaborată interfața serială originală. Acest protocol implică stabilirea conexiunii între două modemuri prin linia telefonică și menținerea fluxului de date dintre acestea cât timp conexiunea este activă. Etapele acestui protocol sunt descrise în continuare. Într-o formă simplificată, acest protocol este utilizat și în cazul comunicației seriale directe între două echipamente, fără utilizarea unor modemuri și a unei linii telefonice.

1. Atunci când un modem aflat la distanță dorește stabilirea conexiunii cu modemul local, transmite semnalul de apel pe linia telefonică. Acest semnal este detectat de către modemul local, care activează semnalul *RI* pentru a informa calculatorul local asupra existenței unui apel telefonic.
2. La detectarea activării semnalului *RI*, pe calculatorul local se lansează în execuție un program de comunicație. Acest program indică disponibilitatea calculatorului de a începe comunicația prin activarea semnalului *DTR*.
3. Atunci când modemul local sesizează faptul că terminalul de date (calculatorul) este pregătit, răspunde la apelul telefonic și așteaptă activarea semnalului purtător de către modemul aflat la distanță. Atunci când modemul local detectează semnalul purtător, activează semnalul *CD*.

4. Modemul local negociază cu modemul aflat la distanță o conexiune cu anumiți parametri. De exemplu, cele două modemi 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. Atunci când modemul sesizează activarea semnalului *RTS*, activează semnalul *CTS* pentru a indica faptul că este pregătit pentru recepția datelor de la calculator.
7. În continuare, datele sunt transferate în ambele sensuri între echipamentele aflate la distanță, pe liniile *TD* și *RD*.
8. Deoarece viteza liniei telefonice este mai redusă decât cea a legăturii dintre calculator și modemul local, bufferul modemului 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 calculatorul poate primi din nou date de la modem, reactivează semnalul *RTS*.
10. La încheierea sesiunii de comunicație, semnalul purtător este dezactivat, iar modemul local dezactivează semnalele *CD*, *CTS* și *DSR*.
11. Atunci când sesizează dezactivarea semnalului *CD*, calculatorul local dezactivează semnalele *RTS* și *DTR*.

Din protocolul descris mai sus, rezultă următoarele:

- Calculatorul trebuie să detecteze activarea semnalelor *DSR* și *CTS* înainte de a transmite date către modem. Dezactivarea oricăruia din aceste semnale va opri, de obicei, fluxul de date de la calculator.
- Modemul trebuie să detecteze activarea semnalelor *DTR* și *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.

3.6.2. 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 și 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. Prima variantă utilizează caracterele de control XON/XOFF, iar a doua variantă utilizează caracterele de control ETX/ACK.

În cazul utilizării variantei XON/XOFF, perifericul transmite caracterul XOFF pentru a indica faptul că bufferul său este plin și transmiterea datelor trebuie oprită de calculator. Acest caracter mai este denumit DC1 (*Device Control 1*) și are codul ASCII 0x13, fiind echivalent cu caracterul Ctrl-S. Caracterul Ctrl-S poate fi introdus și de utilizator la anumite programe de comunicație pentru a opri transmiterea datelor de către un echipament cu care este conectat calculatorul. Atunci când perifericul este pregătit pentru a primi noi date, transmite calculatorului caracterul XON. Acest caracter mai este denumit DC3 (*Device Control 3*) și are codul ASCII 0x11, fiind echivalent cu caracterul Ctrl-Q. La anumite programe de comunicație, introducerea caracterului Ctrl-Q anulează efectul caracterului Ctrl-S.

În cazul utilizării 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 0x03 și este echivalent cu caracterul Ctrl-C. Transmiterea caracterului ACK (*ACKnowledge*) indică posibilitatea reluării transmiterii datelor de către calculator. Acest caracter are codul ASCII 0x06 și este echivalent cu caracterul Ctrl-F.

3.7. Conectori

Porturile seriale pot utiliza unul din două tipuri de conectori. Conectorul DB-25 cu 25 de pini a fost utilizat la calculatoarele din generațiile anterioare. La calculatoarele mai noi 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 obișnuită. Tabelul 3.1 indică numele acestor semnale și 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).

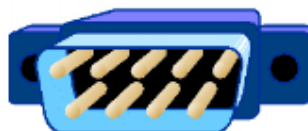


Figura 3.5. Conectorul DB-9 utilizat pentru porturile seriale ale calculatoarelor IBM PC.

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	←

3.8. Cabluri

Există mai multe variante de cabluri care se pot utiliza pentru comunicația serială. Pentru viteze de comunicație reduse și lungimi scurte, se pot utiliza cabluri obișnuite, care nu sunt ecranate. Pentru a reduce interferențele cu alte echipamente, trebuie utilizate cabluri ecranate care conțin un înveliș 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. Conectorul DB-9 nu conține 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 (și a echipamentului), având un rol de protecție. Prin realizarea acestei conexiuni, carcusele metalice ale celor două echipamente conectate prin cablul serial se vor afla la același potențial, evitându-se formarea unor diferențe de tensiune între cele două echipamente, tensiuni care pot fi periculoase pentru acestea. Deseori, conexiunea mesei de protecție lipsește din cablurile seriale.
- Masa de protecție *PG* nu trebuie conectată niciodată la masa electrică *SG*.

Semnalele interfeței seriale au fost prevăzute în scopul 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 același tip (de exemplu, DB-25), este necesar un cablu care conectează pinii cu același număr ai conectorilor de la cele două capete. Acesta este un *cablu 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 celui alt echipament. De aceea, conexiunile acestor pini trebuie inversate la cele două capete ale cablului; un asemenea cablu este numit *cablu inversor*.

3.8.1. Cabluri directe

Într-un cablu direct, fiecare pin corespunzător unui anumit semnal de la un capăt este conectat cu pinul corespunzător aceluiași semnal de la celălalt capăt. În mod frecvent, se utilizează doar o parte a semnalelor interfeței seriale. Chiar în cazul în care fluxul de date este controlat prin metoda hardware, trebuie conectate doar un număr de nouă semnale între un calculator și un modem, presupunând o comunicație asincronă. De exemplu, dacă se utilizează conectori DB-25 la ambele capete ale cablului, pinii care trebuie conectați sunt: 2, 3, 4, 5, 6, 7, 8, 20 și 22. Pentru o comunicație sincronă, sunt necesare două conexiuni suplimentare, reprezentând semnalele de ceas pentru transmisie și recepție, care sunt generate de modemul

sincron. Dacă se utilizează conectori DB-9, toți pinii trebuie conectați, cu excepția 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.

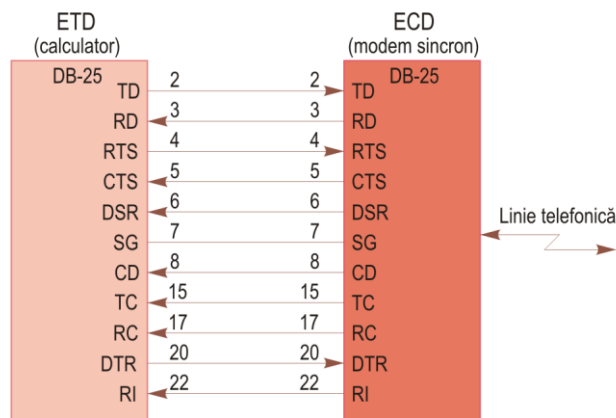


Figura 3.6. Conectarea unui modem sincron la un calculator (conectori DB-25).

3.8.2. Cabluri adaptoare

În cazul în care cei doi conectori ai unei legături seriale sunt de tipuri diferite, de exemplu, DB-25 la un capăt și DB-9 la celălalt capăt, este necesar un cablu adaptor. Deși la calculatoarele IBM PC actuale se utilizează numai conectori DB-9, la unele periferice, ca imprimante seriale, plottere sau modemuri, se utilizează conectori DB-25. Adaptorul poate fi de forma unei cuple având doi conectori de tipuri diferite, sau a unui cablu adaptor având conectori de tipuri diferite la cele două capete.

Tabelul 3.3 prezintă conexiunile necesare pentru un cablu adaptor între un conector DB-25 și un conector DB-9.

Tabelul 3.3. Conexiunile pentru un cablu adaptor între un conector DB-25 și un conector DB-9.

Pin DB-25	Pin DB-9	Semnal	Semnificație
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

3.8.3. Cabluri inversoare

Cablurile inversoare sunt necesare pentru conectarea a două echipamente terminale de date, cum sunt două calculatoare. Aceste cabluri sunt necesare și pentru conectarea unor tipuri de periferice la calculator, cum sunt imprimantele și plotterele seriale, deoarece atunci când a fost concepută interfața serială, aceste periferice au fost considerate ca echipamente terminale de date.

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 sisteme de comunicație serială nu utilizează toate semnalele pentru controlul fluxului de date, astfel încât la aceste sisteme conexiunile pot fi simplificate. 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 și masa electrică. De exemplu, dacă se utilizează conectori DB-25, pinii care trebuie conectați sunt 2, 3 și 7, iar dacă se utilizează conectori DB-9, acești pini sunt 2, 3 și 5. Un asemenea cablu se numește cablu *null-modem*.

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

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

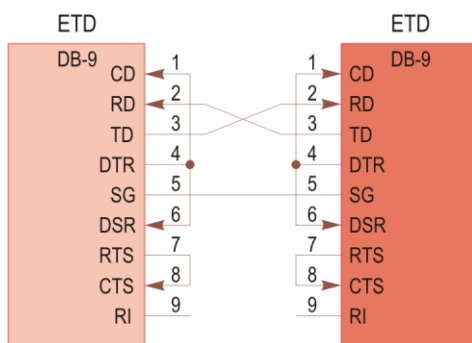


Figura 3.7. Conectarea a două echipamente terminale de date printr-un cablu *null-modem* (conectori DB-9).

Atunci când se utilizează metoda hardware de control a 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 și pinul pentru masa electrică sunt conectați în același fel ca și la cablul *null-modem*. La ambii conectori, pinul *DTR* este conectat cu pinii *DSR* și *CD* de la celălalt capăt al conexiunii. Prin această conexiune, fiecare din cele două echipamente terminale de date poate determina momentul în care celălalt echipament terminal de date este pregătit. Semnalele utilizate pentru controlul fluxului de date sunt *RTS* și *CTS*. La ambii conectori, pinul *RTS* este conectat cu pinul *CTS* de la celălalt capăt al conexiunii. Această conexiune asigură controlul fluxului de date prin metoda hardware, în modul descris în secțiunea 3.6.1.

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 capete.

Observație

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

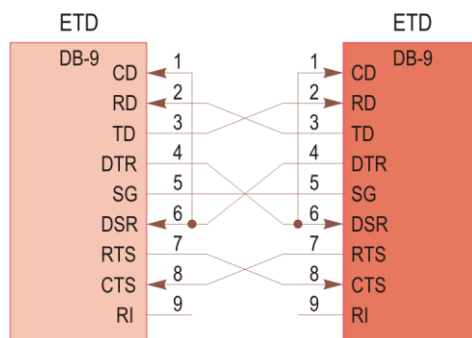


Figura 3.8. Conectarea a două echipamente terminale de date printr-un cablu inversor cu controlul fluxului de date prin hardware (conectori DB-9).

3.9. Circuite UART

Componenta principală a unui port serial este un circuit UART (*Universal Asynchronous Receiver/Transmitter*). Acest circuit realizează conversia datelor paralele de la calculator în formatul necesar pentru transmisia serială și conversia datelor seriale recepționate în formatul paralel utilizat de calculator. Circuitul adaugă bitul de start, bitul de stop și bitul de paritate la datele seriale transmise și detectează acești biți în cadrul datelor seriale recepționate.

La calculatoarele IBM PC și calculatoarele IBM PC/XT originale au fost utilizate circuite UART din familia 8250. Începând cu primele sisteme de 16 biți, a fost utilizat circuitul UART 16450, produs de firma *National Semiconductor*. Acest circuit este compatibil cu circuitele din familia 8250 la nivelul registrelor, dar permite comunicația cu viteze mai ridicate. La calculatoarele IBM PS/2 a fost utilizat circuitul UART 16550, care a fost adoptat și pentru sistemele bazate pe procesorul 80386 și următoarele. Circuitul 16550 funcționează similar cu circuitele 8250 și 16450, dar conține în plus câte o memorie FIFO de 16 octeți pentru transmisie și recepție. Această memorie permite viteze de comunicație superioare față de vitezele 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 îmbunătățite ale circuitului UART 16550 sunt circuitele 16650, 16750 și 16850.

În continuare se descriu principalele caracteristici ale unor circuite UART.

8250

Acest circuit a fost utilizat la primele calculatoare IBM PC. Circuitul nu conținea un buffer de transmisie și nici un buffer de recepție, motiv pentru care viteza de comunicație asigurată era redusă. Circuitul avea mai multe erori de funcționare. Programele ROM BIOS ale calculatoarelor IBM PC și PC/XT au fost realizate astfel încât să țină cont de aceste erori.

8250A

La această versiune, s-au corectat unele erori ale circuitului UART 8250, printre care și o eroare de funcționare a registrului de validare a întreruperilor. Deoarece programele ROM BIOS ale calculatoarelor IBM PC și PC/XT ținneau cont de această eroare, circuitul 8250A nu funcționa corect cu acele calculatoare. Circuitul 8250A funcționa cu calculatoarele IBM PC/AT, dar nu funcționa corespunzător la viteze de 9.600 biți/s și mai mari, din cauza lipsei bufferelor de transmisie și de recepție.

8250B

La această ultimă versiune a circuitelor din familia 8250, s-au corectat erorile întâlnite la cele două versiuni anterioare. Eroarea de funcționare a registrului de validare a întreruperilor din versiunea originală 8250 a fost reintrodusă, pentru ca circuitul să funcționeze în modul așteptat de programele ROM BIOS ale calculatoarelor IBM PC și PC/XT. Nici acest circuit nu funcționa corespunzător la viteze de 9.600 biți/s și mai mari.

16450

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

16550

Reprezintă varianta îmbunătățită a circuitului 16450, conținând buffere de transmisie și de recepție de câte 16 octeți, organizate sub forma unor memorii FIFO. Circuitul permite și transferuri prin canale multiple DMA. Versiunea inițială a acestui circuit nu permitea utilizarea memoriilor FIFO, eroare care a fost corectată în versiunea 16550A. Ultima versiune produsă de *National Semiconductor* este 16550D. Prin utilizarea memoriilor FIFO, vitezele de comunicație pot fi crescute în mod semnificativ, eliminându-se posibilitatea pierderii unor caractere la vitezele mai ridicate. Viteza maximă de comunicație permisă de circuitul 16550 este de 115.200 biți/s.

16650, 16750 și 16850

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

- Circuitul 16650 conține două memorii FIFO de câte 32 octeți;
- Circuitul 16750 conține două memorii FIFO de câte 64 octeți;
- Circuitul 16850 conține două memorii FIFO de câte 128 octeți.

Aceste circuite permit viteze de comunicație superioare, de 230,4 Kbiți/s (16650), 460,8 Kbiți/s (16750) și 921,6 Kbiți/s (16850). Utilizarea acestor circuite este recomandată în cazul unor legături seriale externe de viteză ridicată, cum sunt cele realizate printr-un adaptor ISDN.

3.10. Porturile seriale ale calculatoarelor IBM PC

Sistemul ROM BIOS al calculatoarelor IBM PC originale permitea utilizarea a două porturi seriale, cu numele COM1 și COM2. Ulterior, numărul porturilor a fost extins cu încă două, cu numele COM3 și COM4. Începând cu sistemul de operare *Windows 95*, numărul porturilor seriale a fost extins la 128. Aceste porturi sunt gestionate cu ajutorul driverelor de dispozitiv care le controlează.

Accesul la porturile seriale se poate realiza prin funcții BIOS (întreruperea 0x14), prin funcții 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 patru cuvinte succesive de 16 biți, începând cu adresa 0x0000:0x0400 corespunzătoare portului COM1.

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

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

Port serial	Adresa de bază	Întrerupere
COM1	0x3F8	IRQ 4
COM2	0x2F8	IRQ 3
COM3	0x3E8	IRQ 4
COM4	0x2E8	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 (IRQ 4) cu portul COM1, iar portul COM4 partajează întreruperea de nivel 3 (IRQ 3) cu portul COM2. Magistrala PCI permite partajarea nivelelor de întrerupere, astfel încât pentru plăcile de extensie bazate pe magistrala PCI (sau PCI Express) este posibilă utilizarea unui singur nivel de întrerupere fără apariția unor conflicte.

3.11. Registrele circuitelor UART 16x50

Registrele circuitelor UART sunt accesibile prin instrucțiuni de I/E. Pentru primul port serial, registrele au adrese cuprinse între 0x3F8 și 0x3FF, iar pentru al doilea port registrele au adrese între 0x2F8 și 0x2FF. Primele două adrese permit accesul la mai multe registre ale circuitului. Există anumite 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 și 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.

COM1	COM2	Offset	DLAB	Acces	Abreviere	Denumire
0x3F8	0x2F8	+ 0	0	W	THR	Transmitter Holding Register
			0	R	RBR	Receiver Buffer Register
			1	R/W	–	Divisor Latch Register LSB
0x3F9	0x2F9	+ 1	0	R/W	IER	Interrupt Enable Register
			1	R/W	–	Divisor Latch Register MSB
0x3FA	0x2FA	+ 2	X	R	IIR	Interrupt Identification Register
			X	W	FCR	FIFO Control Register
0x3FB	0x2FB	+ 3	X	R/W	LCR	Line Control Register
0x3FC	0x2FC	+ 4	X	R/W	MCR	Modem Control Register
0x3FD	0x2FD	+ 5	X	R	LSR	Line Status Register
0x3FE	0x2FE	+ 6	X	R	MSR	Modem Status Register
0x3FF	0x2FF	+ 7	X	R/W	–	Scratch Register

Observație

- Circuitele 16650, 16750 și 16850 conțin registre suplimentare față de cele indicate în tabelul 3.5. Structura 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, care este selectat dacă bitul DLAB al registrului LCR este 0. Caracterul care urmează a fi transmis trebuie înscris în acest registru. Dacă registrul TSR (*Transmitter Shift Register*) se golește (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 și se începe transmisia caracterului.

Dacă registrul THR se golește (deci, poate fi înscris cu un nou caracter), circuitul generează o cerere de 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 și se începe transmisia acestuia.

Dacă registrul TSR se golește (deci, poate începe transmisia unui nou caracter), dar registrul THR este gol (sau, dacă memoriile FIFO sunt validate și memoria FIFO de transmisie se golește sub un anumit nivel de prag), circuitul generează o cerere de 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. Prezența unui caracter în registrul RBR se poate determina prin testarea bitului 0 al registrului LSR. Dacă memoriile FIFO sunt invalide, caracterul recepționat trebuie citit de program din registrul RBR înaintea recepției unui nou caracter; în caz contrar, va apare o eroare de suprapunere (*overrun error*). Recepția unui caracter determină generarea unei cereri de întrerupere, 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 unui caracter, conținutul registrului RSR este transferat în mod automat în registrul RBR dacă memoriile FIFO sunt invalidate. Dacă aceste memorii sunt validate, caracterele recepționate sunt depuse în memoria FIFO de recepție.

Divisor Latch Register LSB & MSB

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

$$\text{Divizor} = 1.843.200 / (\text{DebitBinar} * 16)$$

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	0x0900	4.800	0x0018
150	0x0300	9.600	0x000C
300	0x0180	19.200	0x0006
600	0x00C0	38.400	0x0003
1.200	0x0060	57.600	0x0002
2.400	0x0030	115.200	0x0001

IER - Interrupt Enable Register

Reprezintă registrul de validare a întreruperilor, fiind accesibil dacă bitul DLAB al registrului LCR este 0. Circuitul UART poate genera cinci tipuri de cereri de întrerupere, cu nivele de prioritate diferite. Registrul IER permite validarea individuală a acestor întreruperi. Structura acestui registru este ilustrată în figura 3.9.

Biții registrului IER sunt descriși în continuare.

- Biții 7..4 sunt neutilizați (setați la 0).
- Bitul 3 (Enable Modem Status Interrupt) validează prin valoarea 1 generarea unei cereri de întrerupere la modificarea registrului de stare al modemului (MSR).

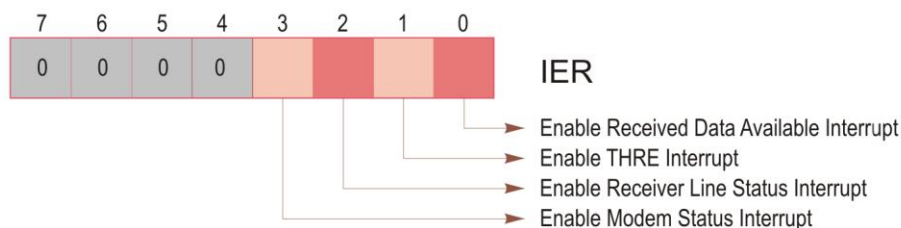


Figura 3.9. Registrul de validare a întreruperilor (IER).

- Bitul 2 (Enable Receiver Line Status Interrupt) validează prin valoarea 1 generarea unei cereri de întrerupere 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 unei cereri de întrerupere 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 unei cereri de întrerupere la recepția unui caracter. Dacă memoriile FIFO sunt validate, acest bit validează și generarea întreruperii de depășire 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. Deși întreruperile generate de circuitul 16x50 apar pe un singur nivel de întrerupere (IRQ 4 pentru primul port serial și 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 acestui registru indică starea memoriilor FIFO.

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

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	Semnalul CTS sau DSR sau RI sau CD s-a modificat	Citirea registrului MSR
0 0 1 0	2	Terminare transmisie caracter	Registrul THR este gol	Citirea IIR sau scrierea unui 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	Citirea registrului RBR sau golirea memoriei FIFO de recepție sub nivelul de prag
1 1 0 0	1	Depășirea timpului (time-out)	Nu s-au citit sau depus caractere din/în memoria FIFO de recepție pe durata a patru caractere și există cel puțin un caracter în memoria FIFO	Citirea registrului RBR
0 1 1 0	0 (maximă)	Modificare stare linie	Eroare de suprapunere, de încadrare sau de paritate, sau întrerupere Break	Citirea registrului LSR

Din tabelul 3.7 rezultă că circuitul UART se poate utiliza și prin interogare software, 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).

La o operație de recepție cu memoriile FIFO validate, circuitul UART va genera o cerere de 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 patru caractere. Aceasta reprezintă întreruperea de depășire a timpului (*time-out*) și a fost prevăzută pentru cazul în care transmițătorul oprește transmisia caracterelor pentru a aștepta un răspuns de la receptor. Fără această întrerupere, ar fi posibil ca receptorul să nu sesizeze faptul că au fost recepționate caractere, deoarece memoria FIFO de recepție nu conține un număr suficient de caractere pentru a se genera o întrerupere de recepție. Combinația de biți din registrul IIR care indică această cauză de întrerupere poate fi ignorat, deoarece circuitul UART va indica și faptul că sunt disponibile caractere în bufferul de recepție.

Biții 7..4 ai registrului IIR sunt descriși 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 erorii de utilizare a memoriilor FIFO.

11: Memoriile FIFO sunt validate și operaționale.

- Biții 5..4 sunt rezervați (setați la 0).

FCR - FIFO Control Register

Reprezintă registrul de control al memoriilor FIFO, fiind disponibil la circuitul UART 16550 și la următoarele. Structura registrului FCR este ilustrată în figura 3.10.

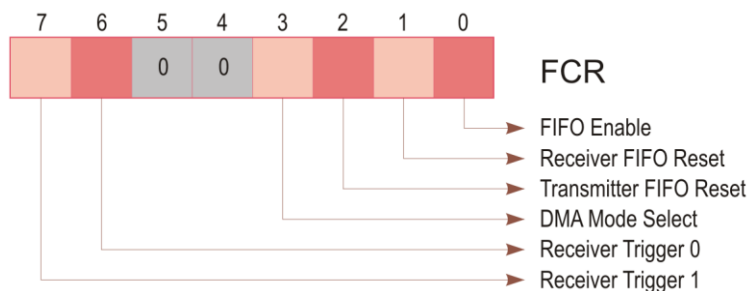


Figura 3.10. Registrul de control al memoriilor FIFO FCR.

Biții registrului FCR sunt descriși 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 cerere de întrerupere după fiecare caracter recepționat, ceea ce va reduce timpul necesar tratării întreruperilor. Semnificația biților 7..6 este următoarea:

00: Cererea de întrerupere se generează după fiecare caracter recepționat;

01: Cererea de întrerupere se generează după 4 caractere recepționate;

10: Cererea de întrerupere se generează după 8 caractere recepționate;

11: Cererea de întrerupere se generează după 14 caractere recepționate.

- Biții 5..4 sunt rezervați (setați la 0).
- Bitul 3 (DMA Mode Select), prezent la circuitul 16550 și următoarele, permite selec-tarea modului de transfer prin DMA. Atunci când memoriile FIFO sunt validate, exis-tă două moduri de transfer prin DMA care pot fi selectate, modul 0 și 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, care sunt executate în mod continuu până când memoria FIFO de recepție se golește sau memoria FIFO de transmisie se umple. La circuitul 16450, este permis doar modul 0.
- Bitul 2 (Transmitter FIFO Reset) permite ștergerea memoriei FIFO de transmisie prin setarea acestui bit la 1.
- Bitul 1 (Receiver FIFO Reset) permite ștergerea memoriei FIFO de recepție prin seta-re a acestui bit la 1.
- Bitul 0 (FIFO Enable) validează prin valoarea 1 memoriile FIFO de transmisie și de recepție. În mod implicit, aceste memorii sunt invalidate, pentru compatibilitate cu circuitele UART 8250 și 16450. Prin setarea acestui bit la 0 se vor invalida memoriile FIFO, datele memorate în acestea fiind pierdute.

Observație

- În cazul în care nu se utilizează memoriile FIFO, sau biții 7..6 ai registrului FCR sunt setați astfel încât să se genereze o cerere de întrerupere după fiecare caracter recepțio-nat, în rutina de tratare întreruperii de recepție este suficient să se citească registrul de recepție RBR o singură dată. Dacă se utilizează memoriile FIFO, în rutina de tratare a întreruperii de recepție trebuie implementată o buclă de program pentru citirea câte unui caracter din registrul de recepție RBR, cât timp există cel puțin un caracter în acest registru. Prezența unui caracter în registrul de recepție RBR este indicată prin faptul că bitul 0 din registrul LSR este setat la 1. După citirea unui caracter din regis-trul RBR, circuitul va încărca 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 comunicației seriale. Structura registrului LCR este ilustrată în figura 3.11.

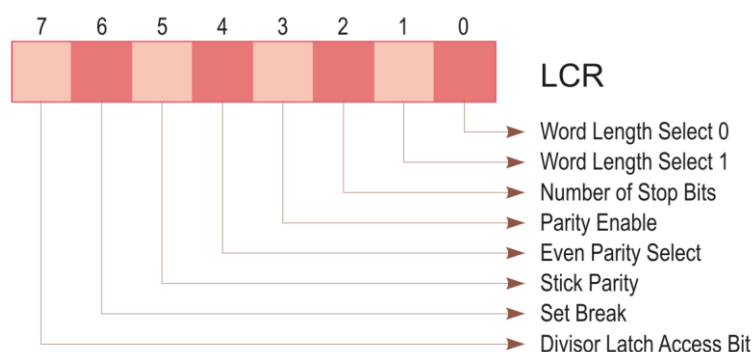


Figura 3.11. Registrul de control al liniei LCR.

Semnificația biților registrului LCR este următoarea:

- Bitul 7 (Divisor Latch Access Bit) modifică funcția registrelor accesibile prin adresele 0x3F8 (0x2F8) și 0x3F9 (0x2F9). Dacă acest bit este 0, registrele accesibile sunt *Transmitter / Receiver Buffer Register*, respectiv *Interrupt Enable Register*. Dacă bi-

tul 7 este 1, registrele accesibile sunt cele utilizate pentru înscrierea divizorului care stabilește debitul binar (LSB, respectiv MSB).

- Bitul 6 (Set Break). Dacă este setat la 1, circuitul forțează linia de comunicație 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 adusă în starea normală prin setarea bitului 6 la 0.
- Bitul 5 (Stick Parity) permite transmiterea sau așteptarea unor biți de paritate cu valoare fixă, 0 sau 1:

0: Paritatea este testată în mod obișnuit, conform biților Parity Enable și 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 și verificarea parității este validată prin bitul Parity Enable:

0: Paritate impară;

1: Paritate pară.

- Bitul 3 (Parity Enable) validează sau invalidează generarea și verificarea parității:

0: Generarea și verificarea parității este invalidată;

1: Generarea și verificarea parității este validată.

- Bitul 2 (Number of Stop Bits) indică numărul biților de stop generați sau așteptați de circuitul UART:

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 controlul comunicației cu modemul.

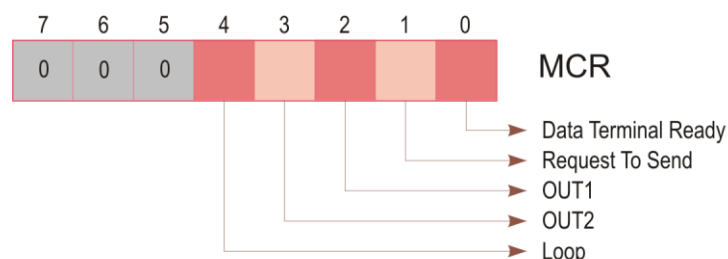


Figura 3.12. Registrul de control al modemului MCR.

Semnificația biților registrului MCR este următoarea:

- Biții 7..5 sunt neutilizați (setați la 0).
- Bitul 4 (Loop) permite testarea circuitului UART și a programelor de comunicație. Prin setarea acestui bit la 1 se vor efectua următoarele operații:
 1. Ieșirea serială a transmițătorului este plasată în starea 1 logic.
 2. Intrarea serială a receptorului este deconectată.
 3. Datele de la ieșirea registrului TSR vor fi recepționate în bufferul de recepție RBR.
 4. Liniile de intrare pentru controlul modemului *DSR*, *CTS*, *RI* și *DCD* sunt deconectate, iar comanda lor se poate realiza cu biții 0..3 ai registrului MCR (Data Terminal Ready, Request To Send, OUT1, respectiv OUT2). Dacă întreruperile sunt validate, modificările acestor biți vor genera cereri de întrerupere ca și în cazul în care semnalele ar fi activate de modem.
- Biții 2 și 3 (OUT1 și 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* și *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).

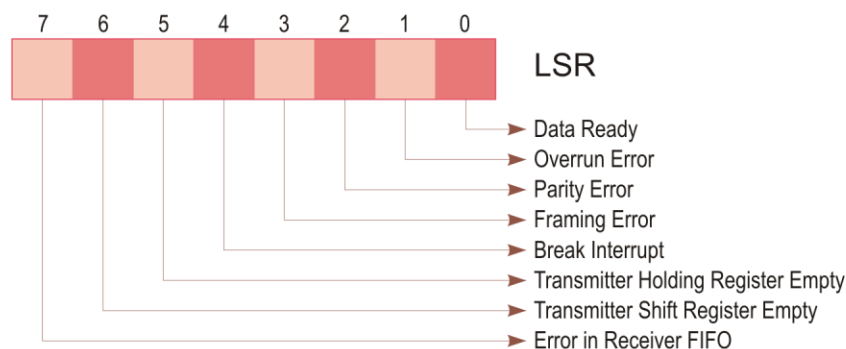


Figura 3.13. Registrul de stare a liniei LSR.

Semnificația biților registrului LSR este următoarea:

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

la 0 atunci când se înscrie un nou caracter în registrul THR. Dacă memoriile FIFO sunt validate, acest bit este setat la 1 atunci când memoria FIFO de transmisie se golește și este resetat la 0 atunci când se înscrie cel puțin un caracter în memoria FIFO de transmisie.

- Bitul 4 (Break Interrupt) este setat la 1 dacă sunt sesizate spații (0 logic) pe linie pentru o perioadă mai mare decât cea necesară pentru transmisia unui caracter. În acest caz, se înscrie un octet cu valoarea 0 în bufferul de recepție și se generează o cerere de întrerupere. Acest bit este resetat prin citirea registrului LSR.
- Bitul 3 (Framing Error) indică o eroare de încadrare, fiind setat la 1 dacă un caracter se recepționează fără biții de stop corespunzători. Receptorul testează numai primul bit de stop, indiferent de numărul biților de stop programați. La detectarea acestei erori, circuitul încearcă să se resincronizeze. Acest bit este resetat la 0 prin citirea registrului LSR.
- Bitul 2 (Parity Error) indică o eroare de paritate, fiind setat la 1 dacă se recepționează un caracter cu paritatea diferită de cea așteptată. Acest bit este resetat la 0 prin citirea registrului LSR.
- Bitul 1 (Overrun Error) indică o eroare de suprapunere. Este setat la 1 dacă se recepționează un nou caracter înaintea citirii caracterului din registrul RBR de către program. În acest caz, se pierde unul sau mai multe caractere. Eroarea de suprapunere, ca și celelalte erori, generează o cerere de întrerupere. Acest bit este resetat la 0 prin citirea registrului LSR. Dacă memoriile FIFO sunt validate și memoria FIFO de recepție se umple peste nivelul de prag, o eroare de suprapunere va fi semnalată doar după ce memoria FIFO s-a umplut și următorul caracter a fost recepționat în registrul RSR.
- Bitul 0 (Data Ready) este setat la 1 atunci când a fost recepționat un caracter și acesta a fost transferat în registrul RBR sau în memoria FIFO. Acest bit este resetat la 0 prin citirea caracterului din registrul RBR sau în urma citirii tuturor caracterelor din memoria FIFO de recepție. Recepția unui caracter va genera o cerere de întrerupere dacă acest tip de întrerupere este validată.

MSR - Modem Status Register

Acest registru conține informații despre starea modemului (figura 3.14). Semnificația biților registrului MSR este următoarea:

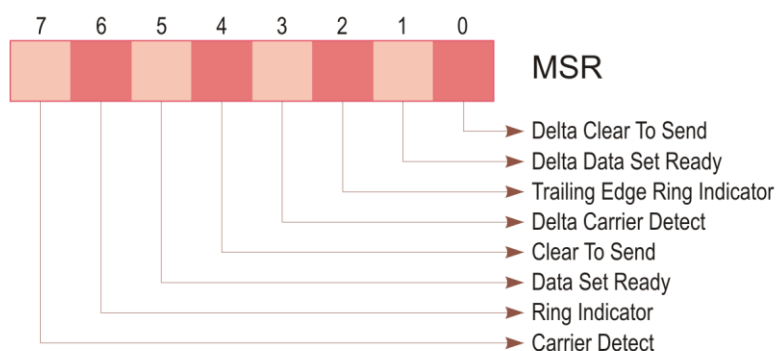


Figura 3.14. Registrul de stare a modemului MSR.

- Biții 7..4 indică starea curentă a semnalelor *CD*, *RI*, *DSR*, respectiv *CTS*. Un semnal activ este indicat prin bitul corespunzător din registrul MSR setat la 1. Dacă bitul Loop al registrului MCR este 1, starea biților 7, 6, 5, 4 din registrul MSR este echivalentă cu starea biților OUT2, OUT1, Data Terminal Ready, respectiv Request To Send din registrul MCR.

- Biții 3..0 indică modificarea stării semnalelor *CD*, *RI*, *DSR*, respectiv *CTS*, de la ultima citire a registrului MSR. Acești biți sunt resetați la citirea registrului MSR.

Observație

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

Scratch Register

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

3.12. Aplicații

3.12.1. Răspundeți la următoarele întrebări:

- a. Care este deosebirea dintre unitatea de măsură a vitezei de modulație și cea a vitezei de comunicație?
- b. Cum se poate asigura sincronizarea între ceasul receptorului și ceasul transmițătorului în cazul unei comunicații 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* și *CTS*?

3.12.2. Scrieți o funcție pentru inițializarea portului serial COM1 cu următorii parametri: debitul binar de 19.200 biți/s, lungimea caracterelor de 8 biți, fără bit de paritate, 1 bit de stop. Utilizați fișierul de definiții pentru portul serial ComDef.h, disponibil pe pagina laboratorului în arhiva ComDef.zip.

3.12.3. Scrieți o funcție care transmite un caracter prin portul serial COM1. Utilizați această funcție pentru scrierea unui program care transmite un șir de caractere prin portul serial COM1. În acest program, apelați funcția scrisă pentru aplicația 3.12.2 pentru inițializarea portului serial. Conectați un cablu serial inversor între porturile COM1 ale două calculatoare. Lansați în execuție programul *HyperTerminal* pe calculatorul utilizat ca receptor și creați o conexiune cu aceiași parametri ca și cei ai portului serial transmițător. Lansați apoi în execuție programul de transmisie și verificați funcționarea acestuia urmărind caracterele afișate în fereastra programului *HyperTerminal*.

3.12.4. Scrieți o funcție care recepționează un caracter prin portul serial COM1. Utilizați această funcție pentru scrierea unui program care recepționează un șir de caractere prin portul serial COM1. Programul va afișa pe ecran fiecare linie recepționată, terminată cu caracterul CR (0x0D), și se va termina atunci când se recepționează caracterul ESC (0x1B). În acest program, apelați funcția scrisă pentru aplicația 3.12.2 pentru inițializarea portului serial. Conectați un cablu serial inversor între porturile COM1 ale două calculatoare. Lansați în execuție programul *HyperTerminal* pe calculatorul utilizat ca transmițător și creați o conexiune cu aceiași parametri ca și cei ai portului serial al calculatorului receptor. Lansați apoi în execuție programul de recepție și verificați funcționarea acestuia introducând linii de text în fereastra programului *HyperTerminal*.

3.12.5. Modificați programul de recepție scris pentru aplicația 3.12.4 astfel încât acesta să transmită înapoi (în ecou) fiecare caracter recepționat la portul serial COM1. Verificați funcționarea programului într-un mod similar cu procedura descrisă la aplicația 3.12.4.

3.12.6. Conectați două calculatoare printr-un cablu serial inversor. Utilizați programul de transmisie scris pentru aplicația 3.12.3 și programul de recepție scris pentru aplicația 3.12.4 pentru transmiterea unui șir de caractere de la unul din calculatoare la celălalt. Calculatorul receptor va afișa pe ecran șirul de caractere recepționat.

Bibliografie

- [1] Baruch, Z., *Sisteme de intrare/ieșire, 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] Wikimedia Foundation, Inc., “RS-232”, Wikipedia, The Free Encyclopedia, 2005, <http://en.wikipedia.org/wiki/RS-232>.