7. INTERFAȚA ATA

Această lucrare de laborator prezintă mai multe variante ale interfeței ATA pentru unitățile de discuri și pune în evidență îmbunătățirile aduse de aceste versiuni interfeței ATA originale. De asemenea, lucrarea descrie adresarea sectoarelor, modurile de transfer ale interfeței, registrele interfeței, lista principalelor comenzi și exemple de comenzi.

7.1. Prezentare generală a interfeței ATA

ATA (*AT Attachment*) este interfața cea mai utilizată pentru conectarea unităților de discuri magnetice la calculatoarele personale. Denumirea "*AT Attachment*" provine de la faptul că interfața a fost proiectată inițial pentru conectarea unei unități de discuri direct la magistrala unui calculator IBM PC/AT (*Advanced Technology*), magistrală numită ISA (*Industry Standard Architecture*) sau AT. ATA este o interfață paralelă de 16 biți. O variantă serială a acestei interfețe, denumită *Serial ATA* (SATA), a fost introdusă în anul 2000 și este utilizată în sistemele de calcul începând din anul 2002.

Interfața ATA este numită și IDE, de la numele primelor unități de discuri cu această interfață. Denumirea IDE (*Integrated Drive Electronics*) se referă la unitățile de discuri care au controlerul integrat în unitate și nu pe o placă separată, ca la interfețele anterioare. Ansamblul format din unitate și controler este conectat la unul din conectorii de pe placa de bază a calculatorului.

Primele unități de discuri care utilizau interfața ATA au fost produse în anul 1986 de firmele Control Data Corporation (CDC), Western Digital (WD) și Compaq, care au stabilit și asignarea semnalelor la pinii conectorului ATA. Pentru eliminarea incompatibilităților și a problemelor legate de interfațarea unităților ATA cu sistemele bazate pe magistrala ISA sau EISA (*Extended ISA*), în anul 1988 a fost înființată comisia *CAM* (*Common Access Method*) a organizației *ANSI*. Această comisie a elaborat prima versiune a standardului interfeței CAM ATA, o versiune de lucru a acestui standard fiind publicată în anul 1989.

Ulterior, specificațiile interfeței paralele ATA au fost elaborate și actualizate de un grup independent care reprezenta principalii producători de calculatoare și unități de discuri, comitetul tehnic T13 (www.t13.org), care a fost o parte a comitetului internațional de standarde în tehnologia informației *INCITS* (*InterNational Committee on Information Technology Standards*). Standardele elaborate de acest comitet au fost aprobate și publicate de institutul american de standarde *ANSI* (*American National Standards Institute*). Același comitet a fost responsabil și cu actualizarea standardelor interfeței ATAPI (*AT Attachment Packet Interface*); această interfață permite conectarea unităților de discuri optice prin aceeași interfață fizică ca și interfața ATA, utilizând însă un protocol logic diferit. Începând cu versiunea a patra a standardului ATA, specificațiile interfeței ATAPI au fost incluse în standardul ATA. Ultima versiune a standardului ATA este *AT Atachment 8*, care a fost publicată în anul 2008. Pentru elaborarea și actualizarea specificațiilor standardului SATA a fost format un grup de lucru separat, numit *Serial ATA Workgroup* (www.serialata.org).

Interfața ATA permite conectarea în serie a două unități de discuri la un conector al interfeței aflat pe placa de bază printr-un cablu cu trei conectori: unul pentru conectarea la placa de bază și două pentru conectarea la unitățile de discuri. Dintre cele două unități, una este unitatea primară (*master*), iar cealaltă este unitatea secundară (*slave*). Fiecare unitate de discuri are propriul controler integrat în unitate, dar cele două unități utilizează aceeași magistrală.

Pentru funcționarea corectă, este necesar ca un singur controler să răspundă la o comandă la un moment dat. De obicei, aceasta se asigură prin poziționarea corespunzătoare a unor comutatoare de pe cele două unități.

Calculatoarele personale conțin două interfețe ATA integrate în setul de circuite, care permit conectarea unui număr de până la patru unități de discuri.

7.2. Evoluția standardelor ATA

De la introducerea versiunii inițiale a standardului ATA, pe măsură ce tehnologia interfețelor ATA din industrie s-a îmbunătățit, au fost elaborate diferite versiuni ale standardelor ATA, care au inclus în specificațiile lor îmbunătățirile apărute anterior. Fiecare versiune a standardului ATA este compatibilă cu versiunile anterioare. Aceasta înseamnă că o unitate de discuri mai veche poate fi utilizată cu o interfață ATA conformă cu o versiune mai nouă a standardului. În general, versiunile mai noi ale standardelor ATA pot fi considerate ca extensii ale versiunilor anterioare.

În continuare sunt descrise principalele caracteristici ale diferitelor versiuni ale standardelor ATA.

ATA (ATA-1)

Versiunea inițială a standardului ATA a fost aprobată oficial de institutul *ANSI* în anul 1994. Această versiune, ca și versiunile ulterioare, specifică o interconexiune paralelă care provine din magistrala ISA (AT) de 16 biți. Standardul a eliminat diferite probleme de incompatibilitate între primele generații de unități de discuri ATA/IDE, în special atunci când două unități de discuri ale unor producători diferiți au fost conectate la aceeași interfață ATA.

Standardul ATA original definește următoarele caracteristici ale interfeței ATA:

- Conectori cu 40 sau 44 de pini;
- Un singur canal ATA, care poate fi partajat de două unități de discuri, configurate ca o unitate *master* și o unitate *slave*;
- Modurile de transfer programat (PIO – *Programmed Input/Output*) 0, 1 și 2, cu caracteristici de sincronizare și rate de transfer diferite;
- Modurile de transfer prin acces direct la memorie (DMA – *Direct Memory Access*) singulare (cu transferuri de un singur cuvânt) 0, 1 și 2;
- Modul de transfer DMA multicuvânt 0;
- Adresare de tip CHS (*Cylinder, Head, Sector*), care specifică numărul cilindrului, al capului și al sectorului de pe unitatea de discuri.

Deși versiunea inițială a standardului ATA permitea o capacitate maximă teoretică a unităților de discuri de 128 GB în binar (137 GB în zecimal[1]), standardul nu specifica modul în care se poate elimina bariera de capacitate de 504 MB (528 MB în zecimal) cauzată de interfața de programare INT 13h a programului BIOS, deoarece, în acel moment, nu existau unități de discuri cu capacitatea mai mare de 504 MB.

ATA-2

ATA-2 reprezintă o extensie a standardului pentru interfața ATA originală, extensie elaborată ca urmare a îmbunătățirilor tehnologice ale unităților de discuri și a cererii crescute a capacității de memorare. Publicat în anul 1996, standardul păstrează compatibilitatea cu interfața ATA originală și aduce îmbunătățiri ale acesteia, fără a fi necesare modificări ale unităților instalate sau ale programelor existente.

---

[1] 1 GB în binar (notat și cu GiB) este egal cu 2^{30} (1.073.741.824) octeți, în timp ce 1 GB în zecimal este egal cu 10^9 (1.000.000.000) octeți.

Principalele îmbunătățiri introduse de standardul ATA-2 sunt următoarele:

- Moduri de transfer PIO mai rapide (modurile PIO 3 și 4);
- Moduri de transfer DMA mai rapide (modurile DMA multicuvânt 1 și 2);
- Comenzi suplimentare care permit transferuri pe blocuri (cuvinte multiple) în scopul creșterii performanțelor;
- Unități de discuri care permit, în mod opțional, adresarea logică pe blocuri (LBA – *Logical Block Addressing*) și interfețe de programare BIOS care realizează translatarea parametrilor CHS, în scopul creșterii capacității adresabile a unităților până la 7,88 GB (8,46 GB în zecimal);
- Comandă *Identify Device* îmbunătățită, care permite unității de discuri să raporteze informații suplimentare necesare pentru sisteme "*Plug and Play*" și pentru compatibilitatea cu reviziile viitoare ale standardului.

Standardul ATA-2 a fost cunoscut sub diferite denumiri care au reprezentat termeni de marketing utilizați de diferite firme, și nu standarde reale. Astfel, firmele Seagate și Quantum au utilizat denumirile Fast ATA și Fast ATA-2 pentru a se referi la diferite porțiuni ale standardului ATA-2. De exemplu, Fast ATA includea transferurile PIO în modul 3 și DMA multicuvânt în modul 1, în timp ce Fast ATA-2 includea în plus transferurile PIO în modul 4 și DMA multicuvânt în modul 2. Ambele variante permiteau transferuri pe blocuri și adresarea logică LBA.

Firma Western Digital a utilizat denumirea EIDE (*Enhanced* IDE) pentru extensia pe care a propus-o standardului ATA-2. Principalele îmbunătățiri au fost introducerea unui canal ATA suplimentar, care utilizează o întrerupere diferită și adrese diferite, și posibilitatea conectării unităților de discuri optice sau a unităților de bandă. Dintre modurile de transfer îmbunătățite specificate de standardul ATA-2, EIDE a inclus transferul PIO în modul 3 sau modul 4 și transferul DMA multicuvânt în modul 1.

ATA-3

Standardul ATA-3, publicat în anul 1997, este o revizie minoră a standardului ATA-2. Această revizie nu a definit noi moduri de transfer cu performanțe mai ridicate. Principalele modificări introduse de standardul ATA-3 sunt următoarele:

- Eliminarea protocoalelor pentru transferurile DMA de un singur cuvânt;
- Adăugarea tehnologiei S.M.A.R.T. (*Self-Monitoring, Analysis, and Reporting Technology*) pentru predicția defectării unităților de discuri;
- Adăugarea unui mod care permite protecția datelor înregistrate pe unitățile de discuri printr-o parolă;
- Specificarea modului de adresare LBA ca fiind obligatoriu (acest mod a fost opțional la standardul ATA-2);
- Recomandări pentru terminarea magistralei la sursă și la destinație în scopul creșterii fiabilității la modurile de transfer cu viteze ridicate.

Tehnologia S.M.A.R.T., dezvoltată inițial de firma IBM, permite sistemului de operare să monitorizeze parametrii de funcționare ai unei unități de discuri în scopul detectării unor degradări ale performanței acesteia. Această degradare se poate accentua în mod progresiv, conducând în final la defectarea unității și la pierderea datelor înregistrate. Prin utilizarea acestei tehnologii este posibilă predicția defectării unității și notificarea utilizatorului astfel încât acesta poate copia datele de pe unitate pe un alt dispozitiv de memorare pentru a preveni pierderea datelor. Tehnologia S.M.A.R.T. nu permite însă predicția defectării subite a unei unități de discuri.

ATA/ATAPI-4

Publicat în anul 1998, standardul ATA/ATAPI-4 a introdus modificări importante ale versiunii anterioare ATA-3. În primul rând, a fost adăugat protocolul ATAPI (ATA *Packet Interface*), care permite conectarea unor periferice cum sunt unitățile de discuri optice la un canal ATA. Conectarea acestor periferice la interfața ATA a fost deja posibilă înaintea versiunii ATA/ATAPI-4 a standardului, dar ATAPI era un standard publicat separat. A fost adăugată o nouă comandă la setul de comenzi ATA, numită *Packet*, care permite transmiterea unei structuri de date cunoscută ca pachet de comandă la un periferic ATAPI. Setul de comenzi recunoscut de perifericele ATAPI este diferit de cel utilizat de interfața ATA, fiind derivat din setul de comenzi al interfeței SCSI. Motivul este că setul de comenzi și setul de registre ATA nu sunt adecvate pentru unele comenzi specifice unităților optice.

A doua modificare importantă introdusă de standardul ATA/ATAPI-4 a fost adăugarea unui nou protocol de transfer numit Ultra-ATA sau Ultra-DMA (UDMA), la care transferurile de date au loc pe ambele fronturi ale semnalului de ceas. Există mai multe moduri de transfer Ultra-DMA, din care specificațiile ATA/ATAPI-4 includ modurile 0, 1 și 2. De exemplu, modul 2 Ultra-DMA permite o rată maximă de transfer de 33,3 MB/s, motiv pentru care acest mod este numit și Ultra-ATA/33 sau UDMA/33. Posibilitatea utilizării unui anumit mod de transfer este condiționată de unitatea de discuri, de setul de circuite de pe placa de bază și de sistemul de operare sau de BIOS.

Principalele îmbunătățiri introduse de standardul ATA/ATAPI-4 sunt următoarele:

- Includerea comenzii *Packet* și a protocolului corespunzător pentru transmiterea comenzilor ATAPI;
- Adăugarea protocolului Ultra-ATA și a modurilor 0, 1 și 2 care utilizează acest protocol, ratele maxime de transfer ajungând la 33,3 MB/s;
- Creșterea integrității datelor prin utilizarea unui cod ciclic redundant (CRC);
- Definirea unui cablu opțional cu 80 de fire (dintre care 40 de fire sunt de masă), care permite creșterea imunității la zgomote;
- Posibilitatea utilizării unui adaptor *Compact Flash* pentru calculatoarele portabile;
- Posibilitatea suprapunerii comenzilor (o nouă comandă poate fi transmisă înainte de terminarea execuției comenzilor precedente) prin implementarea de către perifericele ATA și ATAPI a unor cozi pentru memorarea comenzilor.

ATA/ATAPI-5

Această versiune a standardului ATA a fost aprobată în anul 2000. Principalele modificări introduse de această versiune sunt următoarele:

- Adăugarea modurilor Ultra-DMA 3 și 4; modul 4 permite o rată maximă de transfer de 66 MB/s (acest mod este numit și Ultra-ATA/66 sau UDMA/66);
- Utilizarea cablului cu 80 de fire este obligatorie pentru funcționarea în modul UDMA/66;
- Detectarea automată a cablurilor cu 40 sau 80 de fire;
- Modurile UDMA mai rapide decât UDMA/33 sunt validate numai dacă este detectat un cablu cu 80 de fire.

Modul UDMA/66 permite dublarea ratei de transfer maxime a interfeței prin reducerea timpilor de stabilizare a semnalelor și creșterea frecvenței ceasului. Această frecvență mai înaltă crește posibilitatea unor interferențe între semnale în cazul utilizării cablului ATA cu 40 de fire. Pentru eliminarea interferențelor și a zgomotelor, standardul specifică faptul că utilizarea cablului cu 80 de fire, care a fost definit ca opțional în versiunea ATA/ATAPI-4, este

obligatorie pentru modurile de transfer începând cu UDMA/66. Acest cablu poate fi utilizat și cu unitățile de discuri mai vechi, deoarece conține aceiași conectori cu 40 de pini.

ATA/ATAPI-6

Elaborarea acestei versiuni a standardului ATA a început în anul 2000 și standardul oficial a fost publicat în anul 2002. Principalele îmbunătățiri sau modificări față de versiunea anterioară sunt următoarele:

- Adăugarea modului 5 Ultra-DMA, care permite o rată maximă de transfer de 100 MB/s (acest mod este numit și Ultra-ATA/100 sau UDMA/100);
- Creșterea dimensiunii adreselor logice de la 28 de biți la 48 de biți;
- La adresarea LBA pe 48 de biți, dimensiunea alocată pentru numărul de sectoare transferate printr-o singură comandă a crescut de la 8 biți (256 de sectoare sau 128 KB) la 16 biți (65.536 sectoare sau 32 MB), ceea ce permite transferul mai eficient al fișierelor de dimensiuni mari;
- Unitățile de discuri trebuie să utilizeze adresarea LBA, iar adresarea CHS este declarată învechită;
- Adăugarea unor noi comenzi pentru aplicațiile audio-vizuale (AV – *Audio/Visual*).

Prin extinderea dimensiunii adreselor logice la 48 de biți, numărul de sectoare adresabile a crescut de la 2^{28} la 2^{48} (281.474.976.710.656 sectoare). Astfel, capacitatea maximă adresabilă a unităților de discuri a crescut în mod semnificativ, de la 128 GB (137 GB în zecimal) la 128 PB (petaocteți) sau 144.115.188 GB în zecimal. Această extindere a devenit necesară deoarece discuri cu capacitatea de peste 137 GB au apărut în cursul anului 2001, dar erau disponibile inițial numai cu interfața SCSI, care nu avea aceeași limitare ca și interfața ATA.

ATA/ATAPI-7

Această versiune a standardului ATA a fost publicată în anul 2005. Principala îmbunătățire este adăugarea modului 6 Ultra-DMA (Ultra-ATA/133 sau UDMA/133), care permite o rată maximă de transfer de 133 MB/s. Această versiune a standardului include și specificațiile interfeței seriale SATA-150, cu o rată maximă de transfer de 1,5 Gbiți/s (150 MB/s).

AT Attachment 8

Aceasta este ultima versiune a standardului ATA, publicată în anul 2008. Pentru interfața paralelă ATA, nu sunt introduse moduri îmbunătățite de transfer. Pentru interfața serială SATA, această versiune include specificațiile interfeței SATA-300, cu o rată maximă de transfer de 3 Gbiți/s (300 MB/s).

7.3. Adresarea sectoarelor unităților de discuri ATA

Există două metode principale pentru adresarea sectoarelor unei unități de discuri ATA. Prima metodă este adresarea CHS, la care se specifică trei componente pentru adresarea unui sector: numărul cilindrului (pistei), numărul capului de citire/scriere și numărul sectorului. A doua metodă este adresarea LBA, la care se indică o singură adresă logică specifică sectorului care trebuie adresat. Începând cu versiunea ATA/ATAPI-6 a interfeței ATA/ATAPI, unitățile de discuri trebuie să utilizeze adresarea LBA.

Observație

- Atât adresele CHS cât și adresele LBA reprezintă adrese logice ale sectoarelor. Unitatea de discuri va realiza conversia adresei logice într-o adresă fizică printr-o operație de *translatare*, care este specifică unității.

Adresarea CHS a fost concepută pe baza parametrilor fizici ai unei unități de discuri, deși o adresă CHS reprezintă o adresă logică a unui sector. O asemenea adresă este formată din trei câmpuri: numărul cilindrului, numărul capului și numărul sectorului. Cilindrii sunt numerotați de la 0 până la valoarea maximă permisă de modul de translatare curent, dar numărul maxim nu poate depăși 65.535. Capetele sunt numerotate de la 0 la valoarea maximă permisă de modul de translatare curent, dar valoarea maximă nu poate depăși 15. Sectoarele sunt numerotate de la 1 la valoarea maximă permisă de modul de translatare curent, dar valoarea maximă nu poate depăși 255. Aceste valori maxime au fost alese oarecum arbitrar în momentul elaborării metodei de adresare CHS, considerând că ele sunt suficiente pentru unitățile de discuri din acel moment și pentru cele din generațiile următoare.

La citirea secvențială a datelor de pe unitatea de discuri în modul CHS, procesul începe cu cilindrul 0, capul 0 și sectorul 1. În continuare sunt citite toate celelalte sectoare de pe aceeași pistă; apoi, este selectat capul 1 și se citesc toate sectoarele de pe acea pistă. Citirea de pe cilindrul 0 continuă prin selectarea celorlalte capete, până la ultimul. Apoi se selectează următorul cilindru și procesul se repetă.

În cazul adresării LBA, fiecărui sector de pe unitatea de discuri i se asignează o adresă logică unică. Sectoarele logice ale unității sunt alocate liniar; primul sector adresat în modul LBA (sectorul 0) este același ca primul sector logic adresat în modul CHS (cilindru 0, cap 0, sector 1). Alocarea se continuă până la ultimul sector fizic. Indiferent de modul de translatare CHS curent, adresa LBA a unui sector logic dat nu se modifică. Se poate utiliza următoarea ecuație pentru conversia parametrilor CHS într-o adresă LBA:

$$LBA = ((C \times capete\_pe\_cilindru + H) \times sectoare\_pe\_pistă) + S - 1$$

unde *C, H, S* reprezintă numărul cilindrului, numărul capului, respectiv numărul sectorului, în timp ce *capete_pe_cilindru* și *sectoare_pe_pistă* reprezintă valorile pentru modul de translatare curent.

7.4. Moduri de transfer ale datelor

Specificațiile interfeței ATA/ATAPI definesc două categorii de transferuri ale datelor: transferuri programate (PIO – *Programmed Input/Output*) și transferuri prin acces direct la memorie (DMA – *Direct Memory Access*). Pentru fiecare categorie, sunt definite mai multe moduri de transfer, iar fiecare mod este caracterizat printr-o anumită durată a ciclului. Aceste durate determină ratele maxime de transfer care pot fi obținute.

7.4.1. Moduri de transfer PIO

Modurile de transfer PIO sunt mai puțin eficiente, deoarece pentru fiecare cuvânt transferat procesorul trebuie să execute o secvență de program. Există cinci moduri de transfer PIO, numerotate de la 0 la 4. În funcție de protocolul utilizat, există două tipuri de transferuri PIO: fără confirmare și cu confirmare.

În cazul transferurilor PIO *fără confirmare*, interfața ATA/ATAPI nu are confirmarea faptului că procesorul calculatorului poate accepta datele de la unitatea de discuri. Pentru a minimiza riscul pierderii datelor atunci când procesorul este ocupat cu alte activități în timpul transferului unui bloc de date, aceste transferuri se execută cu o viteză redusă, indiferent de posibilitățile calculatorului. Modurile PIO 0, 1 și 2 utilizează transferuri fără confirmare.

În cazul transferurilor PIO *cu confirmare*, se utilizează semnalul de control *IORDY* al interfeței. Dacă este necesar, unitatea de discuri poate activa acest semnal pentru a extinde durata unui ciclu de transfer și pentru a întârzia interfața. Fără utilizarea acestui semnal, transferul poate fi incorect în modurile PIO rapide. Modurile PIO 3 și 4 utilizează transferuri cu confirmare.

În modul de transfer PIO cel mai lent (modul 0), durata unui ciclu nu poate depăși 600 ns. Într-un singur ciclu se transferă 16 biți (2 octeți). Deci, într-o secundă se transferă $2/600 \cdot 10^9$ octeți, iar rata de transfer maximă teoretică este de 3,33 MB/s. Tabelul 7.1 prezintă modurile PIO și ratele maxime de transfer permise de aceste moduri.

Tabelul 7.1. Modurile de transfer PIO ale interfeței ATA/ATAPI.

Mod	Durata ciclului (ns)	Rata de transfer (MB/s)	Standard
PIO 0	600	3,33	ATA
PIO 1	383	5,22	ATA
PIO 2	240	8,33	ATA
PIO 3	180	11,11	ATA-2, IORDY necesar
PIO 4	120	16,67	ATA-2, IORDY necesar

Primele trei moduri (0, 1 și 2) sunt prezente și în standardul ATA inițial. Modurile PIO 3 și 4 sunt specifice standardului ATA-2 și următoarelor. Aceste moduri utilizează semnalul *IORDY* pentru controlul transferului.

Pentru creșterea eficienței, se utilizează transferuri PIO pe blocuri, care sunt inițiate prin comenzile *Read/Write Multiple*. Prin utilizarea acestor comenzi, se reduce numărul cererilor de întrerupere către calculatorul gazdă.

La interogarea unui controler al unității de discuri prin comanda *Identify Device*, aceasta returnează și informații despre modurile PIO și DMA pe care le poate utiliza. Astfel, biții 7..0 ai cuvântului 64 indică modurile PIO avansate care pot fi utilizate. Dacă bitul 0 al acestui cuvânt este setat la 1, unitatea permite utilizarea modului PIO 3, iar dacă bitul 1 este setat la 1, unitatea permite utilizarea modului PIO 4. Biții 7..2 sunt rezervați.

7.4.2. Moduri de transfer DMA

Transferurile de date care sunt inițiate prin comenzi DMA, cum sunt *Read DMA* și *Write DMA*, diferă de transferurile PIO prin două aspecte:

- Transferurile de date se efectuează printr-un canal DMA;
- Se generează o singură cerere de întrerupere la terminarea comenzii.

Transferurile executate prin acces direct la memorie sunt mult mai eficiente decât transferurile PIO, deoarece procesorul este eliberat de sarcina execuției unei secvențe de program pentru fiecare cuvânt transferat. În plus, procesorul poate executa alte operații în timp ce datele sunt transferate direct între unitatea de discuri și memoria principală.

Interfața ATA/ATAPI permite două tipuri de transferuri DMA: de un singur cuvânt și multicuvânt. În cazul transferurilor de un singur cuvânt, calculatorul inițiază un transfer, selectează cuvântul care trebuie transferat, iar apoi controlerul unității transferă acel cuvânt. Aceste operații trebuie repetate pentru fiecare din cuvintele următoare. Aceste transferuri au o eficiență redusă, motiv pentru care nu mai sunt utilizate.

Modurile de transfer DMA de un singur cuvânt sunt prezentate în tabelul 7.2.

Tabelul 7.2. Modurile de transfer DMA de un singur cuvânt ale interfeței ATA/ATAPI.

Mod	Durata ciclului (ns)	Rata de transfer (MB/s)	Standard
DMA singular 0	960	2,08	ATA
DMA singular 1	480	4,17	ATA
DMA singular 2	240	8,33	ATA

Observație

- Transferurile DMA de un singur cuvânt au fost eliminate din standardul ATA-3 și următoarele.

Modurile de transfer DMA multicuvânt permit obținerea unor performanțe superioare. După ce calculatorul inițiază un transfer, acesta selectează primul cuvânt și ultimul cuvânt al blocului care trebuie transferat, iar apoi controlerul unității transferă toate cuvintele blocului. Tabelul 7.3 prezintă modurile de transfer DMA multicuvânt.

Tabelul 7.3. Modurile de transfer DMA multicuvânt ale interfeței ATA/ATAPI.

Mod	Durata ciclului (ns)	Rata de transfer (MB/s)	Standard
DMA multicuvânt 0	480	4,17	ATA
DMA multicuvânt 1	150	13,33	ATA-2
DMA multicuvânt 2	120	16,67	ATA-2

În funcție de controlul operațiilor de transfer, există două tipuri de transferuri DMA: obișnuite și de tip "*bus mastering*". Transferurile obișnuite sunt executate de controlerul DMA al sistemului. Transferurile DMA ale interfeței ATA/ATAPI sunt de tip "*bus mastering*"; acestea sunt executate de logica interfeței, care preia controlul asupra magistralei și execută transferul.

Cuvântul 63 al blocului de date returnat de comanda *Identify Device* indică modurile DMA multicuvânt care sunt permise de unitatea de discuri și modul care este selectat. Biții 0, 1 și 2 ai acestui cuvânt indică prin valoarea 1 faptul că este permis modul 0, 1, respectiv 2. Biții 8, 9 și 10 ai cuvântului 63 indică prin valoarea 1 faptul că este selectat modul 0, 1, respectiv 2.

Începând cu versiunea ATA/ATAPI-4 a standardului ATA, au fost introduse moduri de transfer DMA mai performante, numite Ultra-DMA sau Ultra-ATA. În aceste moduri, datele sunt transferate la ambele fronturi (crescător și descrescător) ale semnalului de ceas utilizat pentru controlul magistralei de date, astfel că rata de transfer se dublează. În plus, la variantele mai performante ale modurilor Ultra-DMA, frecvența semnalului de ceas este mai ridicată comparativ cu cea de la modurile DMA multicuvânt.

Pentru creșterea fiabilității transferurilor în modurile Ultra-DMA, se utilizează transferuri sincrone în locul transferurilor asincrone. Echipamentul care transmite datele (calculatorul la scriere, unitatea de discuri la citire) generează semnalul de ceas și sincronizează transferul datelor cu semnalul de ceas. Deoarece un singur echipament controlează atât semnalul de ceas, cât și liniile de date, sincronizarea transferurilor este mai precisă.

Modurile Ultra-DMA care au fost introduse de diferitele versiuni ale standardului ATA/ATAPI sunt prezentate în tabelul 7.4.

Tabelul 7.4. Modurile de transfer Ultra-DMA ale interfeței ATA/ATAPI.

Mod	Durata ciclului (ns)	Rata de transfer (MB/s)	Standard
Ultra-DMA 0	240	16,67	ATA/ATAPI-4
Ultra-DMA 1	160	25	ATA/ATAPI-4
Ultra-DMA 2	120	33,33	ATA/ATAPI-4
Ultra-DMA 3	90	44,44	ATA/ATAPI-5
Ultra-DMA 4	60	66,67	ATA/ATAPI-5
Ultra-DMA 5	40	100	ATA/ATAPI-6
Ultra-DMA 6	30	133	ATA/ATAPI-7

Cuvântul 88 al blocului de date returnat de comanda *Identify Device* indică modurile Ultra-DMA care sunt permise de unitatea de discuri și modul care este selectat. Biții 0..6 ai acestui cuvânt indică prin valoarea 1 faptul că este permis modul cu numărul corespunzător poziției bitului respectiv (modul 0 pentru bitul 0, până la modul 6 pentru bitul 6). Biții 8..14 ai cuvântului 88 indică prin valoarea 1 faptul că este selectat modul cu numărul corespunzător poziției bitului respectiv minus 8 (modul 0 pentru bitul 8, până la modul 6 pentru bitul 14).

7.5. Interfața ATA serială

7.5.1. Prezentare generală

De la publicarea primei versiuni de lucru a standardului ATA în anul 1989, interfața ATA a fost îmbunătățită în mod continuu, astfel încât viteza acesteia a crescut de peste 25 de ori față de viteza versiunii inițiale. Cu toate acestea, continuarea îmbunătățirii performanțelor interfeței ATA paralele este dificilă din cauza problemelor specifice unei interfețe paralele, cum sunt interferența electromagnetică între semnale sau dificultatea sincronizării semnalelor.

Soluția la aceste probleme constă în utilizarea unei interfețe seriale, a cărei performanțe pot fi îmbunătățite într-un mod mult mai simplu prin creșterea frecvenței semnalului de ceas.

În anul 2000, firma Intel și mai multe firme producătoare de unități de discuri (APT Technologies, Dell, IBM, Maxtor, Quantum și Seagate Technology) au început elaborarea unei interfețe ATA seriale, denumită *Serial* ATA (SATA). Pentru elaborarea specificațiilor acestei interfețe, a fost format un grup de lucru numit *Serial ATA Working Group*. Prima versiune (1.0) a specificațiilor SATA a fost publicată în anul 2001. Pentru îmbunătățirea acestor specificații, în anul 2002 a fost format grupul de lucru *Serial ATA II Working Group*, care a elaborat versiunea 2.0 a specificațiilor SATA. Ulterior, pentru actualizarea specificațiilor SATA și pentru promovarea acestei interfețe a fost formată o nouă asociație industrială, numită *Serial ATA International Organization* (*SATA-IO*). Informații despre activitatea acestei organizații sunt disponibile la adresa https://sata-io.org. Versiunea 3.0 a specificațiilor SATA a fost publicată în anul 2009, iar versiunea 3.1 în anul 2011.

Versiunea 3.2 a standardului SATA a fost publicată în 2013. Această versiune conține specificațiile pentru interfața SATA Express, care permite conectarea a până la două unități de discuri cu interfața SATA sau a unei unități de discuri cu interfața PCI Express. Interfața PCI Express poate utiliza una sau două legături PCI Express. Viteza crescută a magistralei PCI Express permite optimizarea performanței unităților de discuri cu dispozitive semiconductoare (SSD – *Solid-State Drive*) și a unităților de discuri hibride (SSHD – *Solid-State Hybrid Drive*), care conțin memorii *flash* NAND integrate în unitățile de discuri magnetice. Versiunea 3.2 conține și specificațiile unei plăci numită M.2, cu dimensiuni reduse (lățimea de 22 mm, lungimea de 30, 42, 60, 80 sau 110 mm), care se poate utiliza ca unitate de discuri SSD pentru tablete sau calculatoare *ultrabook*. În plus, versiunea 3.2 conține specificațiile microSSD, care elimină conectorul de pe o unitate de discuri SSD, permițând implementarea unei asemenea unități ca un singur circuit integrat amplasat direct pe placa de bază.

Versiunea 3.3 a standardului SATA, publicată în 2016, conține specificațiile tehnologiei SMR (*Shingled Magnetic Recording*), care permite creșterea capacității unităților de discuri cu aproximativ 25%. Cu această tehnologie, atunci când se scrie o nouă pistă, aceasta se va suprapune parțial peste o pistă scrisă anterior, astfel încât pista scrisă anterior va fi mai îngustă, iar densitatea pistelor va crește. Această metodă a fost aleasă deoarece capetele magnetice de scriere nu pot fi la fel de înguste ca și capetele de citire datorită unor limitări fizice. Versiunea 3.3 conține și a facilitate numită *Power Disable*, care permite calculatorului gazdă să oprească alimentarea unei unități SATA. Versiunea 3.4 a standardului SATA a fost publicată în iunie 2018, iar versiunea 3.5 în iulie 2020. Versiunea curentă a standardului SATA este 3.5a, publicată în martie 2021.

Deși interfața ATA serială diferă fizic de interfața ATA paralelă prin cablul și conectorul utilizat, cele două interfețe sunt compatibile din punct de vedere software între ele. Astfel, programele BIOS, sistemele de operare și programele existente care utilizează interfața ATA paralelă pot fi utilizate fără modificări cu interfața ATA serială. De asemenea, această interfață permite conectarea echipamentelor periferice ATA și ATAPI existente, inclusiv a unităților CD-ROM, CD-RW, DVD și a altor echipamente care pot fi conectate la interfața ATA paralelă.

Figura 7.1 ilustrează simbolul interfeței SATA.

Figura 7.1. Simbolul interfeței ATA seriale.

În timp ce interfața ATA paralelă transferă datele printr-un canal de 16 biți, interfața ATA serială utilizează doar două canale seriale unidirecționale, unul pentru transmisie și unul pentru recepție. Deși datele sunt transmise serial bit cu bit, frecvența semnalului de ceas utilizat pentru transfer este mult mai ridicată decât cea a interfeței ATA paralele. De exemplu, în

cazul modului de transfer UDMA/133 al interfeței ATA paralele, frecvența semnalului de ceas este de 33 MHz, fiind transferate două cuvinte (patru octeți) în fiecare ciclu de ceas, de unde rezultă o rată maximă de transfer de 133 MB/s. În cazul primei versiuni a interfeței SATA, frecvența semnalului de ceas este de 1500 MHz (1,5 GHz). Deoarece un octet este codificat prin 10 biți, rata de transfer corespunzătoare acestei frecvențe este de 150 MB/s, cu 12% mai ridicată față de cea a ultimei versiuni a interfeței ATA paralele. În timp ce este dublarea ratei de transfer a interfeței ATA paralele în viitorul apropiat nu este probabilă, rata de transfer a interfeței SATA-600 a crescut de patru ori față de varianta inițială, deoarece această interfață este disponibilă cu o frecvență a semnalului de ceas de 6 GHz și o rată de transfer maximă de 600 MB/s.

Tabelul 7.5 prezintă tipurile de interfețe SATA.

Tabelul 7.5. Tipuri de interfețe SATA.

Tip interfață	Frecvența (GHz)	Rata de transfer
SATA-150	1,5	1,5 Gbiți/s, 150 MB/s
SATA-300	3	3 Gbiți/s, 300 MB/s
SATA-600	6	6 Gbiți/s, 600 MB/s
SATA Express	8	2x8 Gbiți/s, 2 GB/s

La interfața SATA, biții de date se reprezintă pe linia de transmisie prin codificarea "fără revenire la zero" (NRZ – *No Return to Zero*). Prin această codificare, un bit se reprezintă printr-o schimbare a tensiunii electrice a liniei, și nu printr-un anumit nivel al tensiunii. Se utilizează două nivele de tensiune, iar pentru fiecare bit de 1 din șirul de date există o tranziție de la nivelul actual al tensiunii la celălalt nivel. Nivelul tensiunii rămâne apoi neschimbat până la începutul următorului bit de 1, fără să revină la tensiunea zero.

Interfața SATA utilizează codificarea 8b/10b pentru datele transmise pe linia serială, prin care fiecare octet de date se reprezintă printr-o anumită combinație de 10 biți. Această codificare a fost elaborată inițial de firma IBM la începutul anilor 1980 pentru comunicațiile de date de viteză ridicată. Aceeași codificare este utilizată de numeroase interfețe performante de comunicație serială, inclusiv *Gigabit Ethernet*, *Fibre Channel*, *IEEE 1394* și altele. Unul din scopurile codificării 8b/10b este de a asigura că nu există mai mult de patru biți de 0 (sau biți de 1) transmiși consecutiv. Aceasta este de fapt o variantă a codificării RLL (*Run Length Limited*), care utilizează doi parametri pentru definirea formei de codificare. Acești parametri sunt numărul minim (*run length*) și numărul maxim (*run limit*) de biți consecutivi identici din fiecare octet codificat. Varianta utilizată este RLL 0,4, unde 0 reprezintă numărul minim și 4 reprezintă numărul maxim de biți consecutivi identici din fiecare octet.

De asemenea, codificarea 8b/10b asigură că nu există mai mult de șase sau mai puțin de patru biți de 0 (sau biți de 1) într-un singur octet codificat. Deoarece biții de 0 și de 1 sunt reprezentați pe linia de transmisie prin modificarea tensiunii de pe linie, constrângerea anterioară asigură ca distanțele între tranzițiile tensiunii de pe linie să fie echilibrate. Se obține astfel o încărcare mai echilibrată a circuitelor, crescând fiabilitatea.

Semnalele sunt transmise pe două perechi de linii, în mod diferențial. Una din perechi reprezintă canalul de transmisie, iar cealaltă pereche reprezintă canalul de recepție. Se utilizează nivele reduse de tensiune, de 0,25 V. Semnalele unui canal sunt diferențiale în sensul că, dacă pe una din liniile canalului nivelul tensiunii este de 0,25 V, pe cealaltă linie nivelul tensiunii este de –0,25V. În orice moment, tensiunile de pe cele două linii ale unui canal sunt opuse, iar diferența de tensiune între cele două linii este de 0,5 V. Această diferență de tensiune nu este influențată de zgomote sau alte perturbații externe, ceea ce reprezintă un avantaj important al semnalelor diferențiale.

Interfața SATA utilizează o conexiune punct la punct. Deci, spre deosebire de interfața ATA paralelă, la fiecare port SATA se conectează un singur echipament. Astfel, nu există echipamente înlănțuite și nu sunt necesare setări pentru desemnarea echipamentului *master* și al celui *slave*. Pentru conectarea mai multor echipamente, sunt necesare porturi SATA multiple. De obicei, plăcile de bază sunt echipate cu patru porturi SATA, două porturi primare și două porturi secundare.

Principalele avantaje ale interfeței SATA față de interfața paralelă ATA sunt viteza ridicată, dimensiunile reduse ale conectorilor și cablului, lungimea mai mare a cablului, posibilitatea conectării și deconectării echipamentelor fără întreruperea tensiunii de alimentare. În plus, standardele SATA descriu posibilitatea conectării mai multor echipamente la același port SATA prin utilizarea unui multiplicator de port. În acest fel, conexiunile sunt simplificate și se utilizează mai eficient porturile SATA de pe placa de bază, deoarece acestea pot asigura o rată de transfer suficientă pentru mai multe echipamente. De asemenea, este posibilă realizarea unor sisteme mai complexe de memorare, cum sunt matricele de discuri RAID (*Redundant Array of Independent Disks*), în special dacă există o magistrală PCI Express, care poate asigura rata de transfer necesară. Datorită acestor avantaje, interfața SATA a înlocuit treptat interfața ATA paralelă.

7.5.2. Conectori și cabluri

Interfața SATA necesită un conector de date și un conector de alimentare. În numeroase cazuri, se utilizează conectori combinați (combo), care conțin atât conectorul de date cât și conectorul de alimentare. Conectorul de date conține șapte pini și are dimensiuni reduse, lățimea acestuia fiind de 14 mm. Patru pini se utilizează pentru canalele diferențiale de transmisie și de recepție și trei pini se utilizează pentru conexiunile de masă. Prin utilizarea conexiunilor multiple de masă, care separă cele două canale diferențiale de date, se reduc interferențele electrice dintre aceste canale.

Tabelul 7.6 prezintă semnalele interfeței SATA și asignarea acestora la pinii conectorului de date. Toți pinii de masă au o lungime mai mare față de ceilalți pini, astfel încât vor realiza conexiunea înaintea pinilor de semnal. Aceasta permite conectarea și deconectarea unităților de discuri SATA fără oprirea calculatorului.

Tabelul 7.6. Pinii conectorului de date al interfeței SATA.

Pin	Semnal	Descriere
S1	GND	Masă
S2	A+	Transmisie calculator +
S3	A–	Transmisie calculator –
S4	GND	Masă
S5	B–	Recepție calculator –
S6	B+	Recepție calculator +
S7	GND	Masă

Conectorul de alimentare poate avea dimensiuni standard sau dimensiuni mai reduse (mini-SATA sau micro-SATA). Conectorul de alimentare standard conține 15 pini și furnizează tensiunile de 5 V, 12 V și 3,3 V (deși tensiunea de 3,3 V este utilizată de foarte puține unități de discuri). Fiecare tensiune este furnizată de câte trei pini în paralel pentru a reduce impedanța și pentru a crește curentul. Fiecare pin poate furniza un curent de 1,5 A, astfel încât conectorul poate furniza un curent de până la 4,5 A pentru fiecare din cele trei tensiuni. Conectorul de alimentare conține cinci pini de masă în locul unui singur pin de masă, ceea ce asigură o conexiune la masă cu impedanță redusă. Un pin al conectorului de alimentare poate fi utilizat pentru pornirea eșalonată a unităților de discuri și/sau pentru indicarea activității discului. Dacă acest pin este conectat la masă în conector, motorul unității va porni imediat ce se aplică tensiunea de alimentare. Dacă este lăsat neconectat, unitatea va aștepta până când primește o comandă. Aceasta previne ca mai multe unități să pornească simultan, ceea ce ar putea supraîncărca sursa de alimentare. Pinul este conectat la masă de către unitate atunci când aceasta execută o comandă, astfel încât pinul poate fi conectat la o diodă LED pentru a indica activitatea unității.

Figura 7.2 ilustrează un conector SATA combo standard. Conectorul de date cu 7 pini se află în stânga, iar conectorul de alimentare cu 15 pini se află în dreapta.

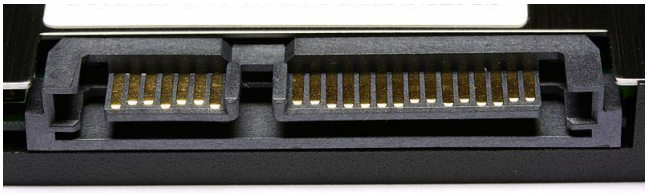

Figura 7.2. Conectorul SATA combo standard al unei unități de discuri.

Un conector combo mini-SATA (mSATA) este destinat dispozitivelor cu dimensiuni mai reduse, cum sunt unitățile optice ale calculatoarelor *notebook*. În acest conector combo, conectorul de date este identic cu conectorul de date al versiunii standard, în timp ce conectorul de alimentare este redus la șase pini. Un pin este prevăzut pentru a semnala prezența dispozitivului, doi pini furnizează în paralel tensiunea de 5 V, doi pini se utilizează pentru conexiunea de masă, iar un pin se utilizează pentru diagnosticare în timpul fabricației. Există adaptoare pasive pentru conversia între un conector SATA standard și un conector mini-SATA. Figura 7.3 ilustrează conectorii unui cablu mini-SATA.

Figura 7.3. Conectorii mini-SATA combo ai unui cablu SATA.

Un conector combo micro-SATA (uSATA) este destinat unităților de discuri de 1,8 inci (46 mm). Conectorul de date al acestui conector combo este similar ca formă cu conectorul de date standard, dar are o grosime mai redusă. Conectorul de alimentare conține nouă pini, dintre care doi pini pentru tensiunea de alimentare de 3,3 V, doi pini pentru tensiunea de alimentare de 5 V și doi pini de masă. Doi pini sunt definiția ca fiind specifici diferiților producători. Figura 7.4 ilustrează conectorii unui cablu micro-SATA.

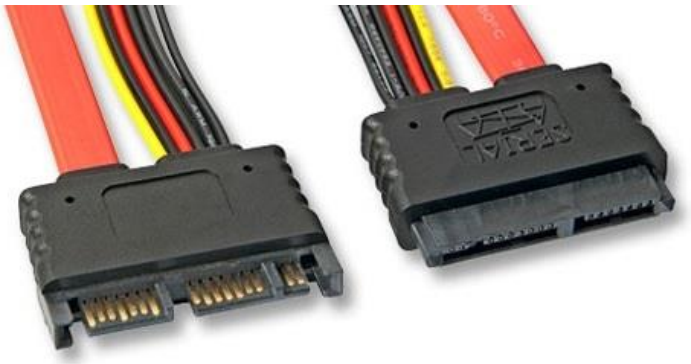

Figura 7.4. Conectorii micro-SATA combo ai unui cablu SATA.

Lungimea maximă a cablului SATA este de 1 m, spre deosebire de cablul interfeței ATA paralele, la care lungimea maximă este de numai 0,45 m. Este necesar un cablu separat pentru conectarea fiecărui echipament la un port SATA al plăcii de bază.

7.5.3. Deosebiri dintre interfețele SATA și SAS

Principalele deosebiri dintre interfețele SATA și SAS (*Serial Attached* SCSI) sunt următoarele:

- Unitățile de discuri SATA sunt identificate prin numărul portului conectat la adaptorul calculatorului gazdă, în timp ce echipamentele SAS sunt identificate prin adresa lor SAS sau *World Wide Name*.
- Spre deosebire de protocolul SATA, protocolul SAS permite existența mai multor inițiatori într-un domeniu SAS.
- Interfața SATA permite conectarea numai a unităților de discuri magnetice și a unităților optice. Interfața SAS permite conectarea și a altor tipuri de echipamente, cum sunt scannere și imprimante. Totuși, aceste echipamente au, de obicei, alte interfețe decât SAS, cum sunt USB, IEEE 1394, Ethernet sau Wi-Fi.
- Interfața SATA utilizează nivele mai reduse de tensiune (0,4 – 0,6 V) decât interfața SAS (0,8 – 1,6 V).
- Din cauza tensiunilor mai ridicate utilizate de interfața SAS, se pot utiliza cabluri mai lungi (până la 8 m) la această interfață, comparativ cu interfața SATA la care lungimea maximă a cablului poate fi de 1 m.
- O deosebire dintre conectorul unei unități SATA și conectorul unei unități SAS este că acesta din urmă are un al doilea set de pini în partea de jos. Acești pini suplimentari sunt prevăzuți pentru al doilea port al unității. Unitățile SATA, care sunt cu un singur port, nu au acești pini.
- Unitățile SATA sunt mai puțin costisitoare decât unitățile SAS și acestea au, de obicei, capacitatea mai ridicată decât cea a unităților SAS.

7.6. Registrele interfeței ATA/ATAPI

Comunicația cu controlerele unităților de discuri se realizează prin registre de I/E. Spre deosebire de alte interfețe, la care numai controlerul selectat recepționează comenzi de la calculator, în cazul interfeței ATA/ATAPI conținutul registrelor se transmite ambelor unități și controlerelor încorporate. Calculatorul realizează distincția dintre cele două unități prin bitul DEV din registrul *Device*. Dacă bitul DEV este 0, este selectată unitatea 0 (*master*), iar în caz contrar este selectată unitatea 1 (*slave*). Dacă există o singură unitate, aceasta trebuie configurată ca *master*.

Datele sunt transferate în paralel (pe 16 biți) între memoria calculatorului și bufferul unității de discuri sub controlul comenzilor transmise în prealabil de la calculator. Datele citite de pe suport sunt memorate în bufferul unității, urmând a fi transferate calculatorului, iar datele transferate din memoria calculatorului sunt memorate în bufferul unității, urmând a fi scrise pe suport. Dacă două unități sunt înlănțuite, comenzile sunt transmise ambelor unități și, cu excepția comenzii de diagnosticare a unităților, numai unitatea selectată va executa comanda.

Tabelul 7.7 prezintă registrele interfeței ATA/ATAPI și deplasamentele lor față de adresa de bază a registrelor din blocul de comandă și adresa de bază a registrelor din blocul de control.

Tabelul 7.7. Registrele interfeței ATA/ATAPI.

Deplasament	ATA		ATAPI	
	Registre din blocul de comandă		**Registre din blocul de comandă**	
	La citire	**La scriere**	**La citire**	**La scriere**
0	Data	Data	Data	Data
1	Error	Features	Error	Features
2	Sector Count	Sector Count	Interrupt Reason	Sector Count
3	LBA Low	LBA Low	LBA Low	LBA Low
4	LBA Mid	LBA Mid	Byte Count Low	Byte Count Low
5	LBA High	LBA High	Byte Count High	Byte Count High
6	Device	Device	Device	Device
7	Status	Command	Status	Command
	Registre din blocul de control		**Registre din blocul de control**	
6	Alternate Status	Device Control	Alternate Status	Device Control

Atunci când interfața ATA implementează adresarea LBA pe 48 de biți, definită începând cu versiunea ATA/ATAPI-6 a standardului, următoarele registre funcționează ca memorii FIFO de câte doi octeți: registrul *Features*, registrul *Sector Count* și registrele de adresă LBA (*Low*, *Mid* și *High*). De fiecare dată când unul din aceste registre este înscris, noul conținut înscris este plasat în locația "înscrisă recent", iar conținutul anterior este mutat în locația "conținutului precedent". De exemplu, atunci când în registrul *Command* se înscrie codul unei comenzi care utilizează adresarea LBA pe 48 de biți, cum este comanda *Read Sector(s) Ext*, adresa utilizată de această comandă și contorul de sectoare sunt indicate în tabelul 7.8.

Tabelul 7.8. Adresa LBA și contorul de sectoare la utilizarea adresării LBA pe 48 de biți.

Registru	Locația "înscrisă recent"	Locația "conținutului precedent"
LBA Low	Adresa LBA, biți 7..0	Adresa LBA, biți 31..24
LBA Mid	Adresa LBA, biți 15..8	Adresa LBA, biți 39..32
LBA High	Adresa LBA, biți 23..16	Adresa LBA, biți 47..40
Sector Count	Contor de sectoare, biți 7..0	Contor de sectoare, biți 15..8

Calculatorul gazdă poate citi locația "conținutului precedent" din registrul *Features*, registrul *Sector Count* și registrele pentru adresa LBA prin setarea la 1 a bitului 7 (HOB – *High Order Bit*) din registrul *Device Control* și apoi citirea registrului dorit. Dacă bitul HOB este 0, la citirea unuia din registrele amintite se citește locația "înscrisă recent". Scrierea se realizează întotdeauna în locația "înscrisă recent", indiferent de starea bitului HOB din registrul *Device Control*.

7.6.1. Registrul de stare *Status*

Acest registru conține starea curentă a unității. Dacă bitul BSY este 0, ceilalți biți ai registrului conțin informații valide; în caz contrar ceilalți biți nu conțin informații valide. Dacă acest registru este citit de calculatorul gazdă în timpul unei întreruperi în curs, condiția de întrerupere este ștearsă.

7	6	5	4	3	2	1	0
BSY	DRDY	DF	#	DRQ	X	X	ERR/CHK

- Bitul 7 (BSY – *Busy*) este setat la 1 ori de câte ori unitatea de discuri are controlul asupra registrelor din blocul de comandă. Dacă bitul BSY este 1, o scriere în oricare registru din blocul de comandă de către calculatorul gazdă va fi ignorată de unitate. Bitul BSY va fi resetat la 0 de către unitate la terminarea execuției unei comenzi și după setarea bitului de stare DRQ la 1 pentru a indica faptul că unitatea este gata pentru transferul datelor.
- Bitul 6 (DRDY – *Device Ready*) este setat la 1 pentru a indica faptul că unitatea de discuri acceptă comenzi. Dacă bitul DRDY este 0, unitatea va accepta și va încerca execuția comenzilor *Device Reset* și *Execute Device Diagnostic*. Alte comenzi nu vor fi acceptate, iar unitatea va seta bitul ABRT din registrul de eroare *Error* și bitul ERR din registrul de stare *Status*, înaintea resetării bitului BSY pentru a indica terminarea comenzii.
- Bitul 5 (DF – *Device Fault*) indică prin valoarea 1 detectarea unui defect de dispozitiv. Un defect de dispozitiv reprezintă orice eveniment care împiedică unitatea de discuri să termine execuția unei comenzi, eveniment care nu este rezultatul unei erori descrise în registrul de eroare *Error*.
- Bitul 4 este specific diferitelor comenzi.
- Bitul 3 (DRQ – *Data Request*) indică prin valoarea 1 faptul că unitatea de discuri este gata pentru transferul datelor între calculatorul gazdă și unitate. După ce calculatorul înscrie codul unei comenzi în registrul de comandă, unitatea setează bitul BSY sau bitul DRQ la 1 până la terminarea comenzii.

- Biții 2..1 sunt nedefiniți.
- Bitul 0 (ERR/CHK – *Error*/*Check*) este definit ca ERR pentru toate comenzile cu excepția comenzilor *Packet* și *Service*, pentru care acest bit este definit ca CHK. Bitul ERR indică prin valoarea 1 apariția unei erori în timpul execuției comenzii precedente. Biții din registrul de eroare *Error* conțin informații suplimentare despre cauza erorii. Bitul CHK indică prin valoarea 1 apariția unei condiții de excepție.

7.6.2. Registrul de date *Data*

Acest registru este de 16 biți și este utilizat pentru citirea sau scrierea datelor în timpul transferurilor de date. Acest registru trebuie accesat pentru transferurile de date în modul PIO numai atunci când bitul DRQ din registrul de stare *Status* este setat la 1.

7.6.3. Registrul de eroare *Error*

Acest registru conține starea ultimei comenzi executate de unitatea de discuri sau un cod de diagnosticare. La terminarea execuției unei comenzi, cu excepția comenzilor *Execute Device Diagnostic* și *Device Reset*, conținutul acestui registru este valid atunci când biții BSY și DRQ din registrul de stare *Status* sunt 0 și bitul ERR/CHK din același registru este 1. La terminarea execuției unei comenzi *Execute Device Diagnostic* sau *Device Reset* și după o resetare hardware sau software, acest registru conține un cod de diagnosticare.

Cu excepția bitului 2 (ABRT), semnificația celorlalți biți ai registrului de eroare *Error* variază în funcție de comanda care a fost executată.

7	6	5	4	3	2	1	0
#	#	#	#	#	ABRT	#	#

- Bitul 2 (ABRT – *Command Aborted*) indică prin valoarea 1 că execuția comenzii cerute a fost abandonată deoarece codul comenzii sau un parametru al comenzii este invalid, comanda nu este implementată sau a apărut o altă eroare.

7.6.4. Registrul *Features*

Acest registru se poate utiliza pentru setarea diferitelor caracteristici ale interfeței, de exemplu, pentru validarea sau invalidarea memoriei *cache* prin comanda *Set Features*. Registrul trebuie înscris numai atunci când biții BSY și DRQ ai registrului de stare *Status* sunt ambii 0. Conținutul acestui registru devine un parametru al comenzii atunci când codul comenzii se înscrie în registrul de comandă *Command*.

Structura acestui registru este specifică diferitelor comenzi. În cazul comenzilor care utilizează adresarea LBA pe 48 de biți, registrul *Features* funcționează ca o memorie FIFO de doi octeți.

7.6.5. Registrul *Sector Count* / *Interrupt Reason*

Pentru interfața ATA, acest registru este numit *Sector Count*. Pentru interfața ATAPI, acest registru este numit *Interrupt Reason*. Registrul trebuie înscris numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru este valid numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru devine un parametru al comenzii atunci când codul comenzii se înscrie în registrul de comandă *Command*.

Structura acestui registru este specifică diferitelor comenzi. În general, acest registru este înscris cu numărul sectoarelor de date care trebuie transferate într-o operație de citire sau scriere între calculatorul gazdă și unitatea de discuri. Pentru unele comenzi, acest registru are un rol diferit de registru contor. Pentru comenzile de acces la suport, acest registru conține valoarea 0 la terminarea comenzii dacă nu au fost erori indicate în registrul de stare. În cazul apariției unor erori, acest registru conține numărul de sectoare care trebuie transferate în scopul terminării operației.

În cazul comenzilor de citire sau scriere care utilizează adresarea LBA pe 28 de biți, dacă registrul *Sector Count* conține valoarea 0x00, aceasta specifică un număr de 256 de sectoare care trebuie transferate. În cazul comenzilor care utilizează adresarea LBA pe 48 de biți, acest registru funcționează ca o memorie FIFO de doi octeți. Pentru aceste comenzi, dacă registrul conține valoarea 0x0000, aceasta specifică un număr de 65.536 sectoare care trebuie transferate.

7.6.6. Registrul *LBA Low*

Acest registru permite înscrierea unei părți a adresei sectorului la comenzile de citire și de scriere care utilizează adresarea LBA. Registrul trebuie înscris numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru este valid numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru devine un parametru al comenzii atunci când codul comenzii se înscrie în registrul de comandă *Command*.

Pentru comenzile care utilizează adresarea LBA pe 28 de biți, registrul *LBA Low* trebuie înscris cu biții 7..0 ai adresei LBA. Pentru comenzile care utilizează adresarea LBA pe 48 de biți, registrul *LBA Low* funcționează ca o memorie FIFO de doi octeți, în modul descris în secțiunea 7.6.

7.6.7. Registrul *LBA Mid* / *Byte Count Low*

Pentru interfața ATA, acest registru este numit *LBA Mid*. Pentru interfața ATAPI, acest registru este numit *Byte Count Low*. Registrul trebuie înscris numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru este valid numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru devine un parametru al comenzii atunci când codul comenzii se înscrie în registrul de comandă *Command*.

Pentru comenzile care utilizează adresarea LBA pe 28 de biți, registrul *LBA Mid* trebuie înscris cu biții 15..8 ai adresei LBA. Pentru comenzile care utilizează adresarea LBA pe 48 de biți, registrul *LBA Mid* funcționează ca o memorie FIFO de doi octeți.

7.6.8. Registrul *LBA High* / *Byte Count High*

Pentru interfața ATA, acest registru este numit *LBA High*. Pentru interfața ATAPI, acest registru este numit *Byte Count High*. Registrul trebuie înscris numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru este valid numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru devine un parametru al comenzii atunci când codul comenzii se înscrie în registrul de comandă *Command*.

Pentru comenzile care utilizează adresarea LBA pe 28 de biți, registrul *LBA High* trebuie înscris cu biții 23..16 ai adresei LBA. Pentru comenzile care utilizează adresarea LBA pe 48 de biți, registrul *LBA High* funcționează ca o memorie FIFO de doi octeți.

7.6.9. Registrul *Device*

Acest registru se utilizează pentru selectarea unității de discuri. Registrul trebuie înscris numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii 0. Conținutul acestui registru este valid numai atunci când bitul BSY din registrul de stare *Status* este 0. Cu excepția bitului DEV, toți ceilalți biți ai acestui registru devin un parametru al comenzii atunci când codul comenzii se înscrie în registrul de comandă *Command*.

7	6	5	4	3	2	1	0
X	LBA	X	DEV	#	#	#	#

- Bitul 7 și bitul 5 sunt nedefiniți.

- Bitul 6 (LBA) selectează modul de adresare al sectoarelor. Anumite comenzi necesită setarea acestui bit la 1 pentru a selecta adresarea LBA. Dacă acest bit este resetat la 0, este selectată adresarea CHS; această adresare a fost utilizată numai până la versiunea ATA/ATAPI-5 a standardului.
- Biții 3..0 sunt specifici diferitelor comenzi.
- Bitul 4 (DEV – *Device Select*) selectează prin valoarea 0 unitatea 0, iar prin valoarea 1 unitatea 1.

7.6.10. Registrul de comandă *Command*

Acest registru conține codul comenzii care trebuie transmis unității de discuri. Execuția comenzii începe imediat după ce codul comenzii este înscris în registrul de comandă *Command*. Conținuturile registrelor din blocul de comandă devin parametri ai comenzii atunci când acest registru este înscris. Scrierea acestui registru șterge orice condiție de întrerupere în curs.

Cu excepția comenzii *Device Reset*, acest registru trebuie înscris numai atunci când biții BSY și DRQ din registrul de stare *Status* sunt ambii egali cu 0.

7.6.11. Registrul *Alternate Status*

Acest registru conține aceleași informații ca și registrul de stare *Status*. Singura deosebire este că citirea registrului de stare alternativă *Alternate Status* nu implică achitarea unei întreruperi sau ștergerea condiției de întrerupere.

7.6.12. Registrul *Device Control*

Acest registru permite calculatorului gazdă să execute o resetare software a unităților de discuri și să valideze sau să invalideze activarea semnalului de întrerupere *INTRQ* de către unitatea selectată. Atunci când registrul *Device Control* este înscris, ambele unități răspund la operația de înscriere indiferent de unitatea care este selectată.

7	6	5	4	3	2	1	0
HOB	X	X	X	X	SRST	nIEN	0

- Bitul 7 (HOB – *High Order Byte*) este definit numai dacă este implementată adresarea LBA pe 48 de biți. Dacă acest bit este setat la 1, citirea registrului *Features*, a registrului *Sector Count* și a registrelor pentru adresa LBA se realizează din locația "conținutului precedent". Dacă bitul HOB este setat la 0, citirea se realizează din locația "înscrisă recent". Scrierea în oricare registru din blocul de comandă are ca efect resetarea bitului HOB la 0.
- Biții 6..3 sunt rezervați.
- Bitul 2 (SRST – *Software Reset*) este bitul de resetare software al unităților de discuri. Dacă există două unități înlănțuite, prin setarea acestui bit la 1 se resetează ambele unități.
- Bitul 1 (nIEN – *INTRQ Enable*) validează prin valoarea 0 activarea semnalului cererii de întrerupere *INTRQ* de către unitatea de discuri.
- Bitul 0 trebuie să fie resetat la 0.

7.7. Protocoale utilizate pentru comenzile ATA

Standardele ATA/ATAPI definesc protocoalele utilizate pentru transferurile de date între calculatorul gazdă și unitatea de discuri și durata ciclurilor de citire/scriere. Această secțiune prezintă exemple de protocoale utilizate pentru execuția comenzilor ATA: protocolul pentru comenzi care nu transferă date, protocolul pentru intrare în modul PIO și protocolul pentru comanda *Execute Device Diagnostic*.

7.7.1. Protocolul ATA pentru comenzi fără transfer de date

Acest protocol este utilizat pentru comenzi cum sunt *Check Power Mode*, *Flush Cache*, *Read Native Max Address Ext*, *Read Verify Sector(s) Ext*, *SMART Enable Operations* și *SMART Return Status*. Presupunând că generarea întreruperilor nu este validată, protocolul ATA pentru comenzile care nu transferă date este următorul:

1. Programul citește registrul de stare *Status* și așteaptă până când biții BSY și DRQ devin 0. Dacă acești biți nu devin 0 într-un anumit timp (de exemplu, 1 secundă), programul poate presupune că unitatea nu răspunde sau nu există și trebuie să abandoneze execuția protocolului.
2. Programul resetează bitul DEV din registrul *Device* la 0 dacă trebuie selectată unitatea 0 sau îl setează la 1 dacă trebuie selectată unitatea 1.
3. Programul inițializează registrele cu parametrii comenzii. Aceste registre pot include registrul *Features*, registrul *Sector Count* și registrele LBA. Registrele care trebuie inițializate depind de fiecare comandă. Exemple de comenzi sunt descrise în secțiunile 7.8.2-7.8.4.
4. Programul scrie codul comenzii în registrul de comandă *Command*. Unitatea va începe execuția comenzii cerute.
5. Programul citește registrul de stare *Status* și așteaptă până când bitul BSY devine 0, ceea ce indică terminarea execuției comenzii. Dacă bitul BSY nu devine 0 într-un anumit timp (de exemplu, 1 secundă), programul trebuie să abandoneze execuția protocolului.
6. Programul testează bitul ERR/CHK din registrul de stare *Status*. Dacă acest bit este 1, comanda s-a terminat cu o eroare; în caz contrar, comanda s-a terminat cu succes.

7.7.2. Protocolul ATA pentru intrare în modul PIO

Acest protocol este utilizat pentru comenzi cum sunt *Identify Device*, *Identify Packet Device*, *Read Buffer*, *Read Sector(s) Ext* sau *SMART Read Log*. Presupunând că generarea întreruperilor nu este validată, protocolul ATA pentru o operație de intrare în modul PIO este următorul:

1. Programul citește registrul de stare *Status* și așteaptă până când biții BSY și DRQ devin 0. Dacă acești biți nu devin 0 într-un anumit timp (de exemplu, 1 secundă), programul poate presupune că unitatea nu răspunde sau nu există și trebuie să abandoneze execuția protocolului.
2. Programul resetează bitul DEV din registrul *Device* la 0 dacă trebuie selectată unitatea 0 sau îl setează la 1 dacă trebuie selectată unitatea 1.
3. Programul inițializează registrele cu parametrii comenzii. Aceste registre pot include registrul *Features*, registrul *Sector Count* și registrele LBA. Registrele care trebuie inițializate depind de fiecare comandă. Exemple de comenzi sunt descrise în secțiunile 7.8.2-7.8.4.
4. Programul scrie codul comenzii în registrul de comandă *Command*. Unitatea va pregăti datele cerute pentru transferul la calculatorul gazdă.
5. Programul citește registrul de stare *Status* și așteaptă până când bitul BSY devine 0. Dacă bitul BSY nu devine 0 într-un anumit timp (de exemplu, 1 secundă), programul trebuie să abandoneze execuția protocolului.
6. Programul testează bitul DRQ din registrul de stare *Status*. Dacă acest bit este 0, unitatea a terminat comanda cu o eroare. În acest caz, protocolul este terminat; programul poate citi registrul de eroare *Error* pentru mai multe informații despre eroarea apărută. Dacă bitul DRQ este 1, programul continuă cu etapa 7.

7. Programul transferă un bloc de date prin citirea registrului de date *Data* cuvânt cu cuvânt. Numărul de cuvinte care trebuie transferate depinde de comanda particulară. De exemplu, comenzile *Identify Device* și *Identify Packet Device* necesită transferul unui număr de 256 de cuvinte de date.
8. Dacă au fost transferate toate blocurile de date ale comenzii respective, comanda s-a terminat cu succes. În caz contrar (dacă mai sunt blocuri de date care trebuie transferate), programul continuă cu etapa 9.
9. Programul așteaptă un timp corespunzător unui ciclu de transfer PIO. De exemplu, acesta poate citi registrul de stare alternativă *Alternate Status*, ignorând rezultatul.
10. Programul continuă cu transferul unui nou bloc de date, de la etapa 5.

7.7.3. Protocolul ATA pentru comanda *Execute Device Diagnostic*

Presupunând că generarea întreruperilor nu este validată, protocolul ATA pentru comanda *Execute Device Diagnostic* este următorul:

1. Programul citește registrul de stare *Status* și așteaptă până când biții BSY și DRQ devin 0. Dacă acești biți nu devin 0 într-un anumit timp (de exemplu, 1 secundă), programul poate presupune că unitatea nu răspunde sau nu există și trebuie să abandoneze execuția protocolului.
2. Programul resetează bitul DEV din registrul *Device* la 0.
3. Programul scrie codul comenzii în registrul de comandă *Command*. Unitatea (sau ambele unități, 0 și 1, dacă sunt prezente) va începe execuția auto-diagnosticării.
4. Programul așteaptă un timp de minim 2 ms.
5. Programul citește registrul de stare *Status* și așteaptă până când bitul BSY devine 0, ceea ce indică terminarea execuției comenzii. Dacă bitul BSY nu devine 0 într-un timp de 6 secunde, programul trebuie să abandoneze execuția protocolului.
6. Programul testează rezultatele execuției comenzii. Registrul de eroare *Error* conține un cod de diagnostic. Registrele LBA și registrul Sector Count conțin o semnătură care se poate utiliza pentru identificarea unităților ATA și ATAPI.

Observație

- Codurile de diagnostic returnate de comanda *Execute Device Diagnostic* sunt indicate în tabelul 7.10, iar semnăturile specifice unităților ATA și ATAPI sunt indicate în tabelul 7.11.

7.8. Comenzi ATA

7.8.1. Lista comenzilor ATA

Tabelul 7.9 prezintă principalele comenzi ATA și registrele care trebuie încărcate cu parametrii comenzilor. Semnificația registrelor este descrisă în continuare.

RF: Registrul *Features*;
RSC: Registrul *Sector Count*;
LBA: Registrele de adresă LBA;
RD: Registrul *Device*.

Caracterul V indică un parametru valid pentru registrul respectiv. Pentru registrul *Device* (RD), V indică utilizarea atât a bitului DEV pentru numărul unității, cât și a biților 27..24 ai adresei LBA, iar D indică faptul că este valid numai bitul pentru numărul unității.

Tabelul 7.9. Lista comenzilor ATA.

Comandă	Cod	RF	RSC	LBA	RD
Check Power Mode	0xE5				D
Device Configuration Freeze Lock	0xB1	0xC1			D
Device Configuration Identify	0xB1	0xC2			D
Device Configuration Restore	0xB1	0xC0			D
Device Configuration Set	0xB1	0xC3			D
Device Reset	0x08				D
Download Microcode	0x92	V	V	V	V
Execute Device Diagnostic	0x90				
Flush Cache	0xE7				D
Flush Cache Ext	0xEA				D
Identify Device	0xEC				D
Identify Packet Device	0xA1				D
Idle	0xE3	V			D
Idle Immediate	0xE1				D
NOP	0x00	V			D
Packet	0xA0	V	V	V	D
Read Buffer	0xE4				D
Read DMA	0xC8		V	V	V
Read DMA Ext	0x25		V	V	D
Read DMA Queued	0xC7	V	V	V	V
Read DMA Queued Ext	0x26	V	V	V	D
Read Multiple	0xC4		V	V	V
Read Multiple Ext	0x29		V	V	D
Read Native Max Address	0xF8				D
Read Native Max Address Ext	0x27				D
Read Sector(s)	0x20		V	V	V
Read Sector(s) Ext	0x24		V	V	D
Read Verify Sector(s)	0x40		V	V	V
Read Verify Sector(s) Ext	0x42		V	V	D
Security Disable Password	0xF6				D
Security Erase Prepare	0xF3				D
Security Erase Unit	0xF4				D
Security Freeze Lock	0xF5				D
Security Set Password	0xF1				D
Security Unlock	0xF2				D
Seek	0x70			V	V
Service	0xA2				D
Set Features	0xEF	V	V	V	D
Set Max Address	0xF9			V	V
Set Max Address Ext	0x37			V	D
Set Multiple Mode	0xC6		V		D
Sleep	0xE6				D
SMART Disable Operations	0xB0	0xD9		V	D
SMART Enable Operations	0xB0	0xD8		V	D
SMART Execute Off-Line	0xB0	0xD4		V	D
SMART Read Log	0xB0	0xD5	V	V	D
SMART Return Status	0xB0	0xDA		V	D
SMART Write Log	0xB0	0xD6	V	V	D
Standby	0xE2		V		D
Standby Immediate	0xE0				D
Write Buffer	0xE8				D
Write DMA	0xCA		V	V	V
Write DMA Ext	0x35		V	V	D
Write DMA Queued	0xCC	V	V	V	V
Write DMA Queued Ext	0x36	V	V	V	D
Write Log Ext	0x3F		V	V	D
Write Multiple	0xC5		V	V	V
Write Multiple Ext	0x39		V	V	D
Write Sector(s)	0x30		V	V	V
Write Sector(s) Ext	0x34		V	V	D

7.8.2. Comanda *Execute Device Diagnostic*

Această comandă determină execuția testelor interne de diagnostic de către unitățile de discuri. Dacă sunt prezente, ambele unități conectate la un canal ATA vor executa comanda, indiferent de unitatea care este selectată. Protocolul pentru această comandă este descris în secțiunea 7.7.3. Nu trebuie inițializat niciun registru înaintea scrierii codului de comandă în registrul de comandă *Command*.

După execuția comenzii, bitul ERR din registrul de stare *Status* va fi resetat la 0. Registrul *Error* va conține un cod de diagnostic de 8 biți. Semnificația codurilor de diagnostic este prezentată în tabelul 7.10. Alte coduri decât 0x01 și 0x81 pot indica informații suplimentare despre eșecul unei unități la execuția testelor.

Tabelul 7.10. Coduri de diagnostic returnate de comanda *Execute Device Diagnostic*.

Cod de diagnostic	Descriere
0x01	Unitate 0: succes; Unitate 1: succes sau nu este prezentă
0x00, 0x02-0x7F	Unitate 0: eșec; Unitate 1: succes sau nu este prezentă
0x81	Unitate 0: succes; Unitate 1: eșec
0x80, 0x82-0xFF	Unitate 0: eșec; Unitate 1: eșec

Registrele *Sector Count*, *LBA Low*, *LBA Mid* și *LBA High* vor conține o semnătură care este specifică unităților ATA. Atunci când comanda *Execute Device Diagnostic* este transmisă unei unități ATAPI, această unitate va returna și ea o semnătură în registrele corespunzătoare ale interfeței ATAPI, semnătură care este specifică unităților ATAPI. Tabelul 7.11 indică semnăturile returnate de unitățile ATA și de unitățile ATAPI.

Tabelul 7.11. Semnăturile unităților ATA și ATAPI.

Registru ATA	Semnătură ATA	Registru ATAPI	Semnătură ATAPI
Sector Count	0x01	Interrupt Reason	0x01
LBA Low	0x01	LBA Low	0x01
LBA Mid	0x00	Byte Count Low	0x14
LBA High	0x00	Byte Count High	0xEB

7.8.3. Comanda *Identify Device*

Această comandă permite calculatorului gazdă preluarea parametrilor unității de discuri. Atunci când recepționează această comandă, unitatea pregătește un bloc de 256 cuvinte cu informații despre unitate: parametrii de funcționare, producătorul, modelul, numărul reviziei, numărul de serie etc. Calculatorul gazdă poate transfera blocul de date prin citirea succesivă a registrului de date *Data*. Protocolul pentru această comandă este protocolul pentru intrare în modul PIO (secțiunea 7.7.2). Cu excepția bitului DEV care trebuie resetat sau setat în registrul *Device*, nu trebuie inițializate alte registre înaintea scrierii codului de comandă în registrul de comandă *Command*.

Unitățile ATA nu vor raporta o eroare după execuția acestei comenzi. Unitățile ATAPI vor seta bitul ABRT în registrul de eroare *Error* și vor plasa semnătura caracteristică a unităților ATAPI în registrele *Interrupt Reason*, *LBA Low*, *Byte Count Low* și *Byte Count High* (tabelul 7.11).

Tabelul 7.12 prezintă semnificația unei părți a cuvintelor de 16 biți care sunt returnate de comanda *Identify Device*. Unii parametri ai unității de discuri sunt definiți ca șiruri de caractere ASCII. Fiecare cuvânt conține două caractere ASCII: primul caracter este conținut în octetul cel mai semnificativ al cuvântului, iar al doilea caracter este conținut în octetul cel mai puțin semnificativ. Unii parametri sunt definiți ca două sau patru cuvinte succesive. Pentru acești parametri, partea cea mai puțin semnificativă este conținută în primul cuvânt.

Tabelul 7.12. Semnificația unor cuvinte returnate de comanda *Identify Device*.

Cuvânt	Semnificație
0	Informații generale de configurație
1	Număr de cilindri logici în modul de translatare CHS implicit
3	Număr de capete logice în modul de translatare CHS implicit
6	Număr de sectoare logice pe pistă în modul de translatare CHS implicit
10-19	Număr de serie (20 de caractere ASCII)
23-26	Revizie firmware (8 caractere ASCII)
27-46	Număr model (40 de caractere ASCII)
54	Număr de cilindri logici în modul de translatare CHS curent
55	Număr de capete logice în modul de translatare CHS curent
56	Număr de sectoare pe pistă în modul de translatare CHS curent
57-58	Capacitate în sectoare în modul de translatare CHS curent
60-61	Număr total de sectoare adresabile (adresare LBA pe 28 de biți)
100-103	Număr total de sectoare adresabile (adresare LBA pe 48 de biți)

7.8.4. Comanda *Read Native Max Address Ext*

Această comandă este implementată de unitățile de discuri care permit adresarea LBA pe 48 de biți. Comanda returnează adresa LBA nativă maximă a unității de discuri. Această adresă este adresa cea mai mare acceptată de unitate în condițiile implicite de fabrică. Pentru această comandă trebuie utilizat protocolul pentru comenzi fără transfer de date (secțiunea 7.7.1). Înaintea înscrierii codului comenzii în registrul de comandă *Command*, bitul LBA din registrul *Device* trebuie setat la 1 pentru a specifica adresarea LBA, iar bitul DEV din același registru trebuie resetat sau setat pentru a specifica unitatea selectată.

Adresa nativă maximă de 48 de biți este returnată în registrele *LBA Low*, *LBA Mid* și *LBA High*. Atunci când se utilizează adresarea pe 48 de biți, registrele LBA funcționează ca memorii FIFO de câte doi octeți. Cei doi octeți ai fiecărui registru LBA sunt selectați individual prin resetarea la 0 sau setarea la 1 a bitului HOB din registrul *Device Control*. Tabelul 7.13 indică conținutul registrelor LBA după execuția acestei comenzi.

Tabelul 7.13. Conținutul registrelor LBA după execuția comenzii *Read Native Max Address Ext*.

Registru		Conținut
LBA Low	HOB = 0	Adresa LBA nativă maximă (7..0)
	HOB = 1	Adresa LBA nativă maximă (31..24)
LBA Mid	HOB = 0	Adresa LBA nativă maximă (15..8)
	HOB = 1	Adresa LBA nativă maximă (39..32)
LBA High	HOB = 0	Adresa LBA nativă maximă (23..16)
	HOB = 1	Adresa LBA nativă maximă (47..40)

Dacă această comandă nu este implementată, unitatea va seta la 1 bitul ABRT în registrul de eroare *Error*. În acest caz, și bitul ERR din registrul de stare *Status* va fi setat la 1.

7.8.5. Comanda *SMART Return Status*

Această comandă permite calculatorului gazdă să recepționeze starea de fiabilitate S.M.A.R.T. a unității de discuri. Pentru această comandă trebuie utilizat protocolul pentru comenzi fără transfer de date (secțiunea 7.7.1). Înaintea înscrierii codului comenzii în registrul de comandă *Command*, bitul DEV din registrul *Device* trebuie resetat sau setat pentru a specifica unitatea selectată. În plus, trebuie inițializate următoarele registre: registrul *Features* trebuie setat la 0xDA; registrul *LBA Mid* trebuie setat la 0x4F; registrul *LBA High* trebuie setat la 0xC2.

Pentru unitățile care implementează tehnologia S.M.A.R.T., fiecare producător definește un set de atribute sau parametri operaționali ai unității și setează valori de prag peste care aceste atribute nu trec la funcționarea normală. Exemple de atribute sunt: contorul sectoarelor realocate (atunci când este găsit un sector defect, conținutul acestuia este transferat pe un sector de rezervă); numărul blocurilor de date cu erori necorectabile; contorul încercărilor de a ajunge la viteza de rotație nominală a unității (dacă o primă încercare a eșuat). O condiție

de depășire a pragului înseamnă că cel puțin un atribut a trecut valoarea de prag. Ca răspuns la comanda *SMART Return Status*, unitatea va indica dacă a detectat o condiție de depășire a pragului.

Dacă unitatea nu a detectat o condiție de depășire a pragului, aceasta setează registrul *LBA Mid* la valoarea 0x4F și registrul *LBA High* la valoarea 0xC2. Dacă unitatea a detectat o condiție de depășire a pragului, aceasta setează registrul *LBA Mid* la valoarea 0xF4 și registrul *LBA High* la valoarea 0x2C. Dacă unitatea nu implementează această comandă, dacă facilitatea S.M.A.R.T. este dezactivată sau dacă valorile registrelor de intrare sunt invalide, unitatea va seta la 1 bitul ABRT în registrul de eroare *Error*. În acest caz, și bitul ERR din registrul de stare *Status* va fi setat la 1.

7.9. Controlerele SATA Intel

Componenta *Platform Controller Hub* (PCH) a seturilor de circuite Intel actuale conține două controlere SATA, ambele pe magistrala PCIe 0. Primul controler reprezintă dispozitivul PCIe 31, funcția 2, iar al doilea controler reprezintă dispozitivul PCIe 31, funcția 5 (numărul dispozitivului și funcției poate depinde de setul de circuite). Depinzând de configurația sistemului, poate fi validat doar primul controler sau pot fi validate ambele controlere. Controlerele interacționează cu unitățile de discuri printr-o interfață la nivel de registre care este echivalentă cu cea a unui adaptor tradițional ATA (IDE) al calculatorului gazdă. Cele două controlere permit utilizarea a până la șase porturi SATA. Fiecare port poate fi validat sau invalidat în mod independent și are un controler DMA separat.

Facilitățile oferite de controlerele SATA depind de seria setului de circuite. De exemplu, controlerele SATA ale setului de circuite Intel din seria 8 utilizat de calculatoarele din laborator permit rate de transfer de 6 Gbiți/s, adresarea LBA pe 48 de biți și trei moduri de funcționare: ATA (IDE), AHCI (*Advanced Host Controller Interface*) și RAID (*Redundant Array of Independent Disks*).

Pentru modul de funcționare ATA (IDE), fiecare controler SATA conține un set de registre care păstrează copia registrelor de interfață ATA. Controlerele SATA emulează comportamentul registrelor din blocul de comandă, a registrelor din blocul de control, transferurile de date PIO, transferurile de date DMA și întreruperile.

Controlerele SATA oferă suport hardware pentru interfața de programare AHCI, care a fost dezvoltată de Intel împreună cu alte firme. Această interfață de programare definește structuri de memorie pentru execuția tranzacțiilor între un controler SATA și un driver software, permițând performanțe și facilități avansate. Exemple de facilități avansate permise de interfața de programare AHCI sunt lipsa desemnării unităților SATA ca unități *master*/*slave* (fiecare unitate este tratată ca o unitate *master*), o coadă nativă de comenzi asistată prin hardware (unitatea poate reordona comenzile pentru a crește eficiența transferurilor de date) și posibilitatea conectării și deconectării unităților de discuri fără notificarea prealabilă a sistemului.

Registrele de configurație PCI ale controlerelor SATA Intel cuprind registrele antetului de configurație PCI și un număr de registre specifice PCI. Aceste registre pot fi accesate utilizând fie mecanismul de configurație compatibil PCI, fie mecanismul de configurație îmbunătățit PCIe. Registrele antetului de configurație PCI sunt enumerate în tabelul 7.14. Deplasamentul indicat în tabel este relativ la adresa de bază a spațiului de configurație alocat pentru un anumit controler (dispozitivul 31, funcția 2 pentru primul controler SATA, sau dispozitivul 31, funcția 5 pentru al doilea controler SATA).

Tabelul 7.14. Registrele antetului de configurație PCI ale controlerelor SATA Intel.

Deplasament	Mnemonica	Nume registru	Dimens. (biți)
0x00	VID	Vendor Identification	16
0x02	DID	Device Identification	16
0x04	PCICMD	PCI Command	16
0x06	PCISTS	PCI Status	16
0x08	RID	Revision Identification	8
0x09	PI	Programming Interface	8

Deplasament	Mnemonica	Nume registru	Dimens. (biți)
0x0A	SCC	Sub-Class Code	8
0x0B	BCC	Base Class Code	8
0x0D	PMLT	Primary Master Latency Timer	8
0x0E	HTYPE	Header Type	8
0x10	PCMD_BAR	Primary Command Block Base Address	32
0x14	PCNL_BAR	Primary Control Block Base Address	32
0x18	SCMD_BAR	Secondary Command Block Base Address	32
0x1C	SCNL_BAR	Secondary Control Block Base Address	32
0x20	BAR	Legacy Bus Master Base Address	32
0x24	ABAR/SIDPBA	AHCI Base Address / SATA Index Data Pair Base Address	32
0x2C	SVID	Subsystem Vendor Identification	16
0x2E	SID	Subsystem Identification	16
0x34	CAP	Capabilities Pointer	8
0x3C	INT_LN	Interrupt Line	8
0x3D	INT_PN	Interrupt Pin	8

Registrele antetului de configurație PCI ale controlerelor SATA au aceleași funcții ca și registrele generale ale antetului de configurație PCI descrise în lucrarea de laborator *Magistrala PCI Express*. Registrul PCMD_BAR conține pe pozițiile 15..3 adresa de bază a spațiului de I/E alocat pentru registrele din blocul de comandă ale canalului primar ATA, iar registrul SCMD_BAR conține pe pozițiile 15..3 adresa de bază a spațiului de I/E alocat pentru registrele din blocul de comandă ale canalului secundar ATA. Registrul PCNL_BAR conține pe pozițiile 15..2 adresa de bază a spațiului de I/E alocat pentru registrele din blocul de control al canalului primar ATA, iar registrul SCNL_BAR conține pe pozițiile 15..2 adresa de bază a spațiului de I/E alocat pentru registrele din blocul de control al canalului secundar ATA.

7.10. Aplicații

7.10.1. Răspundeți la următoarele întrebări:

a. Care este scopul tehnologiei S.M.A.R.T.?

b. Care sunt îmbunătățirile introduse în versiunea ATA/ATAPI-6 a standardului ATA?

c. Care sunt avantajele interfeței ATA seriale comparativ cu interfața ATA paralelă?

d. Care sunt îmbunătățirile introduse în versiunile 3.2 și 3.3 ale standardului SATA?

7.10.2. Creați o aplicație *Windows* pentru determinarea adreselor de bază ale registrelor de I/E pentru primul controler SATA al calculatorului. Ca model pentru aplicația *Windows*, utilizați aplicația AppScroll disponibilă pe pagina laboratorului în arhiva AppScroll.zip. Executați următoarele operații pentru a crea proiectul aplicației:

1. În mediul de programare *Visual Studio 2022*, creați un nou proiect gol de tip *Windows Desktop* cu utilitarul *Windows Desktop Wizard*.
2. Verificați că platforma activă a soluției este setată la x64.
3. Modificați proprietatea *Character Set* a proiectului la *Not Set*.
4. Copiați în directorul proiectului fișierele din arhiva AppScroll.zip și adăugați la proiect aceste fișiere.
5. Copiați în directorul proiectului fișierele Hw.h și Hw64.lib din directorul unui proiect creat anterior. Copiați în directorul proiectului fișierele PciBaseAddressUEFI.cpp și ATA-ATAPI.h, disponibile pe pagina laboratorului în arhiva ATA-ATAPI.zip.
6. Adăugați la proiect fișierele Hw.h, ATA-ATAPI.h și PciBaseAddressUEFI.cpp.
7. Specificați fișierul Hw64.lib ca o dependență suplimentară pentru linkeditor.
8. În fișierul sursă AppScroll.cpp, adăugați o directivă `#include` pentru a include fișierul antet ATA-ATAPI.h. Declarați `PciBaseAddressUEFI()` ca o funcție fără parametri care returnează o valoare de tip `DWORD64`.

9. Selectați *Build* → *Build Solution* și urmăriți ca aplicația să fie construită fără erori.

În fișierul sursă AppScroll.cpp, apelați mai întâi funcția `PciBaseAddressUEFI()` pentru a determina adresa de bază a spațiului de configurație extins PCIe și memorați adresa de bază returnată într-o variabilă globală. Dacă funcția returnează 0, adresa de bază nu se poate determina cu succes, iar în acest caz aplicația trebuie să revină cu un cod de eroare. Scrieți apoi o funcție care returnează într-un cuvânt dublu (`DWORD`) adresa de bază a registrelor din blocul de comandă și adresa de bază a registrelor din blocul de control pentru un controler SATA al calculatorului. Funcția are patru parametri de intrare; primii trei parametri sunt numărul magistralei, numărul dispozitivului și numărul funcției PCIe pentru controlerul SATA. Ultimul parametru reprezintă canalul ATA pentru care trebuie returnate adresele de bază: dacă parametrul este 0, funcția trebuie să returneze adresele de bază pentru canalul ATA primar (canalul 0) al controlerului SATA, iar dacă parametrul este 1, funcția trebuie să returneze adresele de bază pentru canalul ATA secundar (canalul 1) al controlerului SATA. În această funcție, apelați funcția care returnează un pointer la antetul de configurație al unei funcții PCIe, scrisă pentru aplicația 2.7.2 din lucrarea de laborator *Magistrala PCI Express*, și accesați registrele adreselor de bază prin acest pointer. Funcția returnează un cuvânt dublu conținând adresa de bază a registrelor din blocul de comandă în cuvântul inferior și adresa de bază a registrelor din blocul de control în cuvântul superior.

Observații

- În registrele PCMD_BAR și SCMD_BAR, adresele de bază ale registrelor din blocul de comandă se află în cuvintele inferioare ale acestor registre; biții 2..0 ai acestor cuvinte trebuie resetați la 0 înaintea returnării adreselor de bază.
- În registrele PCNL_BAR și SCNL_BAR, adresele de bază ale registrelor din blocul de control se află în cuvintele inferioare ale acestor registre; biții 1..0 ai acestor cuvinte trebuie resetați la 0 înaintea returnării adreselor de bază.

După scrierea funcției, includeți un apel al acestei funcții în funcția `AppScroll()` pentru a determina adresele de bază pentru canalul ATA primar al primului controler SATA, și un alt apel pentru a determina adresele de bază pentru canalul ATA secundar al aceluiași controler. Pentru fiecare canal, afișați adresa de bază a registrelor din blocul de comandă și adresa de bază a registrelor din blocul de control.

7.10.3. În fișierul sursă AppScroll.cpp, scrieți o funcție pentru transmiterea comenzii *Execute Device Diagnostic* unei unități de discuri. Parametrul de intrare al funcției este adresa de bază (de tip `WORD`) a registrelor din blocul de comandă pentru un canal ATA; funcția nu returnează nicio valoare. Această comandă este descrisă în secțiunea 7.8.2, iar protocolul pentru această comandă este descris în secțiunea 7.7.3. Funcția afișează mesaje dacă unitatea nu răspunde (biții BSY și DRQ nu devin 0 într-un timp de aproximativ 1 s) sau execuția comenzii nu se termină într-un timp de 6 s. În cazul în care comanda se termină cu succes, funcția citește registrul de eroare *Error* și afișează codul de diagnostic returnat de unitate. În plus, funcția citește registrele conținând semnătura unității (tabelul 7.11) și afișează dacă unitatea este o unitate ATAPI.

După scrierea funcției, includeți un apel al acestei funcții în funcția `AppScroll()` utilizând ca parametru adresa de bază a registrelor din blocul de comandă pentru canalul ATA primar, iar apoi un alt apel utilizând ca parametru adresa de bază a registrelor din blocul de comandă pentru canalul ATA secundar.

Observații

- Definiți simbolic deplasamentele registrelor utilizate în cadrul funcției și măștile de biți necesare pentru aceste registre cu directive `#include` la începutul fișierului sursă (AppScroll.cpp).
- Adresa unui registru se formează adunând deplasamentul său la adresa de bază transmisă funcției ca parametru.

7.10.4. Extindeți aplicația 7.10.3 prin scrierea unei funcții pentru transmiterea comenzii *Read Native Max Address Ext* unei unități de discuri. Parametrii de intrare ai funcției sunt următorii: adresa de bază (de tip `WORD`) a registrelor din blocul de comandă pentru canalul ATA la care este conectată unitatea; adresa de bază (de tip `WORD`) a registrelor din blocul de control pentru canalul ATA la care este conectată unitatea; numărul unității (0 sau 1). Funcția nu returnează nicio valoare. Această comandă este descrisă în secțiunea 7.8.4, iar protocolul pentru această comandă este descris în secțiunea 7.7.1. Funcția afișează mesaje dacă unitatea nu răspunde (biții BSY și DRQ nu devin 0 într-un timp de aproximativ 1 s) sau execuția comenzii nu se termină într-un timp de aproximativ 1 s. În cazul în care comanda se termină cu succes, funcția resetează la 0 bitul HOB din registrul *Device Control*, citește conținutul registrelor LBA și le memorează în variabile locale. Apoi, funcția setează la 1 bitul HOB din registrul *Device Control*, citește din nou conținutul registrelor LBA și le memorează în alte variabile locale. În continuare, funcția determină și afișează adresa LBA maximă a unității de discuri utilizând conținutul registrelor LBA memorate anterior; conținutul registrelor LBA după execuția comenzii este prezentat în tabelul 7.13. În sfârșit, funcția calculează și afișează capacitatea maximă în GB a unității de discuri, presupunând dimensiunea sectorului de 512 octeți.

După scrierea funcției, includeți un apel al acestei funcții în funcția `AppScroll()` utilizând ca parametri adresele de bază ale registrelor din blocul de comandă și ale registrelor din blocul de control pentru canalul ATA primar și numărul unității 0.

7.10.5. Extindeți aplicația 7.10.2 prin scrierea unei funcții pentru transmiterea comenzii *Identify Device* unei unități de discuri. Parametrii de intrare ai funcției sunt adresa de bază (de tip `WORD`) a registrelor din blocul de comandă pentru canalul ATA la care este conectată unitatea și numărul unității (0 sau 1). Funcția nu returnează nicio valoare. Această comandă este descrisă în secțiunea 7.8.3, iar protocolul pentru această comandă este descris în secțiunea 7.7.2, fiind necesar transferul unui singur bloc de date de 256 cuvinte. Funcția afișează mesaje dacă unitatea nu răspunde (biții BSY și DRQ nu devin 0 într-un timp de aproximativ 1 s) sau execuția comenzii nu se termină într-un timp de aproximativ 1 s. Atunci când comanda se termină cu succes, funcția afișează următoarele informații despre unitatea de discuri: numărul modelului, numărul de serie, revizia *firmware*, numărul total de sectoare adresabile cu adresarea LBA pe 28 de biți și numărul total de sectoare adresabile cu adresarea LBA pe 48 de biți. Apoi, utilizând numărul total de sectoare adresabile și presupunând că un sector conține 512 octeți, funcția calculează și afișează capacitatea în GB a unității de discuri pentru adresarea LBA pe 28 de biți și pentru adresarea LBA pe 48 de biți.

După scrierea funcției, includeți un apel al acestei funcții în funcția `AppScroll()` utilizând ca parametri adresa de bază a registrelor din blocul de comandă pentru canalul ATA primar și numărul unității 0.

7.10.6. Extindeți funcția scrisă pentru aplicația 7.10.5 prin afișarea următoarelor informații suplimentare despre unitatea de discuri:

- Numărul maxim de sectoare care pot fi transferate pe întrerupere la execuția comenzilor *Read/Write Multiple* (cuvântul 47, biții 7..0) și setarea curentă pentru numărul de sectoare transferate pe întrerupere (cuvântul 59, biții 7..0);
- Posibilitatea utilizării modului DMA multicuvânt 2 (cuvântul 63, semnificația biților este descrisă în secțiunea 7.4.2);
- Posibilitatea utilizării modului PIO 4 (bitul 1 al cuvântului 64 este setat la 1 dacă se poate utiliza acest mod);
- Durata minimă a ciclului în modurile de transfer PIO cu utilizarea semnalului *IORDY* (valoarea cuvântului 68 reprezintă durata minimă a ciclului în ns);
- Versiunea standardului ATA cu care este compatibilă unitatea de discuri (dacă unul din biții 4, 5, 6, 7 ai cuvântului 80 este setat la 1, unitatea este conformă cu versiunea ATA/ATAPI-4, ATA/ATAPI-5, ATA/ATAPI-6, respectiv ATA/ATAPI-7);

- Posibilitatea adresării LBA pe 48 de biți (bitul 10 al cuvântului 83 este setat la 1 dacă se poate utiliza adresarea pe 48 de biți);
- Posibilitatea utilizării modului Ultra-DMA 6 și modul Ultra-DMA care este selectat (cuvântul 88, semnificația biților este descrisă în secțiunea 7.4.2).

Bibliografie

[1] American National Standards Institute, Inc., "Information Technology - AT Attachment 8 - ATA/ATAPI Architecture Model (ATA8-AAM)", T13/1700-D Revision 3, 2006.

[2] American National Standards Institute, Inc., "Information Technology - AT Attachment 8 - ATA/ATAPI Command Set (ATA8-ACS)", T13/1699-D Revision 4a, 2007.

[3] American National Standards Institute, Inc., "Information Technology - AT Attachment 8 - ATA/ATAPI Parallel Transport (ATA8-APT)", T13/1698-D Revision 2, 2007.

[4] American National Standards Institute, Inc., "Information Technology - AT Attachment 8 - ATA/ATAPI Serial Transport (ATA8-AST)", T13/1697-D Revision 1, 2007.

[5] Intel Corporation, "Intel 8 Series/C220 Series Chipset Family Platform Controller Hub (PCH)", Datasheet, May 2014.

[6] Mueller, S., *Upgrading and Repairing PCs*, 16th Edition, Que Publishing, 2005.

[7] Serial ATA International Organization, "SATA-IO Introduces New Standard for Embedded SSDs", 2011.

[8] Serial ATA International Organization, "SATA-IO Unveils Revision 3.2 Specification", 2013.

[9] Serial ATA International Organization, "What Is SATA Express?", 2015.

[10] The PC Guide, "Integrated Drive Electronics / AT Attachment (IDE/ATA) Interface", 2004.

[11] Wikimedia Foundation, Inc., "SATA", 2024, https://en.wikipedia.org/wiki/SATA.

[12] Wikimedia Foundation, Inc., "SATA Express", 2024, https://en.wikipedia.org/wiki/SATA_Express.