UNIVERZA V MARIBORU
FAKULTETA ZA ELEKTROTEHNIKO,
RAČUNALNIŠTVO IN INFORMATIKO

Drago Fišer

Univerzalni vmesnik za dostop do naprav SCSI

Diplomska naloga

Maribor, marec 2008

UNIVERZA V MARIBORU

FAKULTETA ZA ELEKTROTEHNIKO, RAČUNALNIŠTVO IN INFORMATIKO
2000 Maribor, Smetanova ul. 17

Diplomska naloga univerzitetnega študijskega programa

Univerzalni vmesnik za dostop do naprav SCSI

Študent: **Drago FIŠER**
Študijski program: **univerzitetni, Računalništvo in informatika**
Smer: **Programska oprema**

Mentor: **red. prof. dr. Damjan ZAZULA**

Maribor, marec 2008

UNIVERZA V MARIBORU

FAKULTETA ZA ELEKTROTEHNIKO, RAČUNALNIŠTVO IN INFORMATIKO
2000 Maribor, Smetanova ul. 17

Številka:

Datum:

SKLEP O DIPLOMSKI NALOGI

1. **Drago FIŠER**, absolvent visokošolskega študija Računalništva in informatike, izpolnjuje pogoje za izdelavo diplomske naloge.

2. **Tema diplomske naloge**

   MENTOR: red. prof. dr. Damjan ZAZULA
   SOMENTOR:

3. **Naslov diplomske naloge**

   Univerzalni vmesnik za dostop do naprav SCSI

4. **Vsebina diplomske naloge**

PREDSTOJNIK INŠTITUTA DEKAN

ZAHVALA

Zahvaljujem se mentorju dr. Damjanu Zazuli za pomoč, nasvete in trud pri oblikovanju diplomskega dela. Zahvalil bi se še podjetju Hermes SoftLab, še posebej sodelavcem v projektni skupini OmniBack, ki so me podpirali pri študiju in mi z nasveti in literaturo pomagali k hitrejšemu napredovanju. Nenazadnje velja omeniti še moje najbližje, ki so me vedno znova vzpodbujali in mi dajali potrebno energijo.

Univerzalni vmesnik za dostop do naprav SCSI

Ključne besede: SCSI, gonilnik, API

UDK vrstilec:

Povzetek

Razvijalci programske opreme se redno srečujejo s časovnimi pritiski pri izdaji svojih izdelkov. Želja vsakega proizvajalca je pravočasno izdati kakovostne programe ter jih učinkovito vzdrževati. Pomembna lastnost učinkovitega programa je njegova modularnost, kar je še posebej očitno, če se program prevaja in izvaja na različnih sistemih. Aplikacijski vmesnik (API) je tisti del, ki pripomore k reševanju težav sistemskih neenakosti in naredi progam bolj modularen.

.

V diplomski nalogi smo razvili aplikacijski vmesnik za dostop do naprav, priključenih na serijski vmesnik za majhne računalniške sisteme – SCSI (Small Computer Serial Interface). Aplikacijski vmesnik omogoča sistemsko neodvisne klice funkcij za pošiljanje ukazov SCS- napravam, priključenim na računalniški sistem. Vmesnik premosti razlike, ki jih operacijski sistemi vnesejo s svojimi gonilniki v verigo aplikacijskih klicev.

Vmesnik se je izvrstno izkazal v programskem paketu HP Data Protector in v njega vnaša splošno in hkrati učinkovito metodo za dostopanje do naprav SCSI.

Universal interface for accessing the SCSI devices

Keywords: SCSI, driver, API

UDK:

Abstract

Software developers have constant pressure for regular releases of their products. Every publisher wants to release their products on time and, at the same time, preserve quality. An important attribute of good software is its modularity, in particular when a program is to be built and executed on various platforms. Application interface (API) is a tool that help overcome platform differences and makes a piece of software more modular.

In this diploma thesis we developed an API that allows a software program to access devices connected by Small Computer Serial Interface – SCSI. Application interface enables SCSI calls that are platform independent, and overcomes differences induced by various operating systems having unique driver implementations.

.

Application interface is built in software package HP Data Protector and has proven robust yet generalized enough to allow easy field maintenance. SCSI access has never been that easy within a package before.

KAZALO VSEBINE

KAZALO SLIK

UPORABLJENE KRATICE

SCSI – Small Computer Serial Interface
LUN – Logical Unit Number
API – Application Interface
CDB – Command Descriptor Block
ASC – Additional Sense Code
ASCQ – Additional Sense Code Qualifier
HBA – Host Bus Adapter
SAM – System Administration Manager
SMIT – System Management Interface Tool

1. UVOD

Osnovni cilj in politika večine programskih organizacij je pravočasno izdati kakovostne programske proizvode in jih učinkovito vzdrževati. Za razvoj programske opreme in njeno vzdrževanje je pomembna modularnost programja, ki omogoča neodvisno ter lokalizirano vzdrževanje obstoječe kode na računalniškem sistemu. Če se koda razvija za več sistemov naenkrat, je sistem transparentnosti ključen pri naknadnem vzdrževanju obstoječe kode. Aplikacijski vmesnik (API) je orodje, s katerim se premostijo težave sistemskih neenakosti ter definira skupno dostopno točko za modularnost aplikacije.

Za dostop do naprav, priključenih na računalniški sistem, se uporablja standardni vmesnik SCSI (*Small Computer Serial Interface*) ali serijski vmesnik za majhne računalniške sisteme. Naprave, ki komunicirajo z vmesnikom SCSI, so sistemsko neodvisne ter ne potrebujejo posebnih vmesnikov za njihov priklop, ampak se lahko uporabi standardni vmesnik za vse sisteme.

Operacijski sistemi komunicirajo z vmesnikom SCSI preko gonilnikov. Gonilnik za vmesnik SCSI je enota, ki prevaja sporočila, kot jih razume operacijski sistem, v sporočila, ki jih razume vmesnik SCSI.

Uporabniška aplikacija, ki uporablja naprave, priključene na vmesnik SCSI, in ki želi komunicirati s temi napravami, za komunikacijo prav tako potrebuje gonilnik za naprave SCSI. V tej točki se prekine transparentnost dostopa do naprav SCSI, kajti gonilniki so sistemsko odvisni in jih ima zato vsak operacijski sistem zasnovane po svojih potrebah.

V tej točki se pojavi potreba po sistemsko neodvisnem modulu, ki bi omogočal aplikaciji enovit dostop do naprav SCSI, ne glede na to, v katerem sistemskem okolju se izvaja. Izdelati tak modul je zastavljena naloga te diplomske naloge.

Za izvedbo naloge je prvi korak preučiti standard SCSI. Standard je prisoten že vrsto let in je dobro usidran na področju upravljanja perifernih naprav. Proizvajalci standard dobro poznajo, zato

njihove naprave ustrezajo specifikacijam in praktično ni večjih odstopanj od specifikacij, določenih v standardu. Gonilniki se lahko držijo splošnih pravil delovanja standarda, kar jim omogoča, da generično podpirajo skoraj vse naprave SCSI. Standard SCSI bomo na kratko razložili v prvem poglavju.

Naslednji korak pri izdelavi naloge je preučiti delovanje gonilnikov na izbranih sistemih. Čeprav gonilniki praviloma zakrivajo zasnovo standarda SCSI, se določene podrobnosti temu zakrivanju izmuznejo (npr. statusni zlog) in je poznavanje standarda zelo priročno.

Pri upravljanju naprav z ukazi SCSI je koristno imeti enovit pristop do izvajanja teh ukazov, zato pomeni definirati ta pristop ključni del rešitve. Izbrani pristop smo imenovali SCSIpt in ga podrobneje opisujemo na začetku drugega poglavja. Zasnova omogoča izdelavo takšnega programskega vmesnika, ki ni sistemsko odvisen in odpre možnost enovite izbire klicev za ukaze SCSI.

Nekateri klici za ukaze SCSI so razloženi v drugem poglavju. Standard opredeljuje klice, ki jih določeni tipi naprav morajo podpirati. Tako mora naprava za direkten dostop nujno podpirati ukaza WRITE in READ, naprava za zaporedni dostop zraven že omenjenih tudi REWIND, da omenimo le nekatere. Ti klici uporabljajo strukturo SCSIpt, ki smo jo zasnovali pri izdelavi sistemsko neodvisnega vmesnika.

Pri izbiri klicev ukazov smo se osredotočili na ukaze za naprave z direktnim dostopom (diski), naprave s sekvenčnim dostopom (tračne enote) in naprave za izmenjavo medijev. Določen je tudi splošen klic za izvajanje ukaza SCSI, posredovanega v ukazni vrstici, ki pa v aplikacijskem vmesniku še ni implementiran.

2. Vmesnik SCSI

Serijski vmesnik za majhne računalnike – SCSI je lokalno vhodno-izhodno vodilo, ki zmore velik obseg podatkovnih prenosov [1]. Cilj vmesnika je opremiti računalnike z neodvisnostjo od fizičnih lastnosti naprav iz določenega razreda. Tako so lahko različne naprave, kot so trdi diski, tiskalniki, CD-ROM-i, tračne enote, magnetno-optični diski in druge naprave dodane h konfiguraciji brez potrebe po spremembah na osnovnem vmesniku ali programju. Dopuščena je možnost dodajanja specialnih značilnosti in funkcij, ukazov in polj, rezerviranih za posebne potrebe nekaterih produktov. Določeno število ukazov in polj je rezerviranih za prihodnje razširitve zasnovanega protokola.

Drugi cilj vmesnika je premakniti logiko, ki je specifična za določene naprave, iz naprav SCSI. Nabor ukazov, ki je definiran v protokolu, dopušča obsežen operacijski sistem, ki opravi vse potrebno za inicializacijo priklopljene naprave SCSI. Formalno zaporedje povpraševanj določi tip in karakteristike naprave SCSI ter parametre nastavitev. Dodatna povpraševanja se lahko nanašajo na pripravljenost naprave za delovanje, tipe medijev in druge značilnosti, ki jih naprava podpira. Ti parametri, ki sicer niso nujni za delovanje operacijskega sistema, so vodeni v napravi sami in niso znani vmesniku SCSI.

Protokol SCSI nudi podporo priključitvi več iniciatorjev (pobudnikom za operacijo) naenkrat, prav tako kot več napravam, ki se na njih odzivajo. V arhitekturo vmesnika SCSI je vgrajena distribuirana arbitraža, to je logika, pri kateri naprave tekmujejo za dostop do vodila. Vodilo je dodeljeno napravi z najvišjo prioriteto. Čas za dodeljevanje vodila ni odvisen od števila priključenih naprav in je lahko manjši od 10 μs.

Arbitraža je dodana zato, da je lahko prisotnih več iniciatorjev in da dovoljuje sočasne vhodno/izhodne operacije. Vse naprave SCSI morajo podpirati definirani asinhroni protokol. Zraven tega je definiran še sinhroni protokol in protokol za prenos sporočil. Ukazi so razvrščeni med obvezne, opcijske in specifične za določeno napravo. Vse naprave morajo podpirati obvezne ukaze, ki imajo dovolj funkcionalnosti, da omogočajo obstoj splošnih gonilnikov, ki ne poznajo natančnih lastnosti določenega tipa naprav (npr. kapacitete diska).

2.1. Značilnosti

2.1.1. Fizične in električne značilnosti

Naprave SCSI so med seboj zaporedno povezane s 50-žilnim A-kablom ali 68-žilnim B-kablom. Obe strani povezave morata biti zaključeni s terminatorjem. Vsi signali so med seboj enaki. Na sistemih, kjer je implementiran razširjen (*wide*) SCSI, se naprave priklopijo na kabel tipa B.

Definirani sta dve možnosti glede na povezavo. Prva je enojna (*single-ended*), ki dovoljuje razdaljo prenosa do 6 metrov. Za ta tip povezave hiter sinhroni prenos ni priporočljiv. Drugi tip je diferencialna povezava, ki je namenjena za razdalje do 25 metrov.

2.1.2. Vodilo

Komunikacija na vodilu je dovoljena samo med dvema priključenima napravama naenkrat. Na enem vodilu je lahko priključenih največ 8 naprav (16 v razširjeni verziji). Vsaka naprava ima določeno svojo identifikacijsko številko, ki se imenuje SCSI ID. Možne so vse kombinacije povezav pod pogojem, da je ena izmed naprav, ki komunicirajo, iniciator in se je druga na iniciatorja sposobna odzvati. Iniciator proizvaja operacijo, ki jo ciljna naprava izvede. Ponavadi so naprave samo iniciatorji ali pa samo izvajajo operacije, vendar pa je mogoče tudi, da ena naprava združuje obe funkcionalnosti.

Nekatere funkcije na vodilu so določene iniciatorjem in nekatere ciljnim napravam. Iniciator z arbitražo doseže dostop do vodila in določi svojo ciljno napravo. Ciljna naprava lahko nato zahteva prenos ukaza, podatkov ali drugih informacij na podatkovnem vodilu. V nekaterih primerih lahko tudi arbitrira za vodilo in si izbere drugega iniciatorja, tako da lahko nadaljuje z želeno operacijo. Prenosi podatkov na vodilu delujejo asinhrono, kar pomeni da sledijo protokolu REQ/ACK.

2.1.3. Logične značilnosti

Arhitektura SCSI vključuje osem ločenih faz delovanja. Te so:

splošne faze:

- vodilo prosto,
- arbitraža za vodilo,
- izbira,
- ponovna izbira

ter faze za prenos informacij:

- izstavi ukaza,
- prenos podatkov,
- status prenosa in
- faza sporočila.

2.1.3.1. Faze delovanja

V teh fazah se določita iniciator in ciljna naprava, s katero bo iniciator komuniciral.

Vodilo prosto

V tej fazi vodilo miruje. Ne dogaja se nobena vhodno/izhodna operacija. Naprave SCSI prepoznajo to fazo po signalih na vodilu, ki jih določi (sprosti) ciljna naprava. Iniciatorji ponavadi ne pričakujejo te faze po sproščanju ciljne naprave, razen v naslednjih primerih:

- zgodil se je reset;
- ciljna naprava je zaznala sporočilo ABORT, DISCONNECT, RELEASE RECOVERY, ABORT TAG ali CLEAR QUEUE;
- ciljna naprava je oddala sporočilo COMMAND COMPLETE.

Arbitraža za vodilo

V tej fazi se vodilo enemu izmed iniciatorjev, ki tekmuje za dostop do vodila. Postopek za dodelitev poteka, kot sledi:

- naprava mora najprej počakati na prosto vodilo (faza prostega vodila);
- naprava pošlje signal BSY za zahtevo vodila in svoj SCSI ID;
- po določenem času naprava preveri, ali je na vodilu kakšen signal za zahtevo vodila, ki ima za SCSI ID večje število kot naprava sama. Če se taksen signal zazna, potem je arbitraža izgubljena in se mora naprava vrniti v stanje čakanja na prosto vodilo;
- v kolikor je arbitraža dobljena, je vodilo rezervirano za uporabo in naprava postane iniciator. To naznani s posebnim signalom SEL na vodilu.

Faza izbire

V tej fazi si iniciator izbere svojo ciljno napravo z namenom, da ji poda določeno nalogo. Iniciator pošlje na vodilo SCSI ID za napravo, s katero želi komunicirati, in to naznani s signalom (ATN). Ciljna naprava sporoči, da je rezervirana za uporabo, tako da na vodilu izda signal BSY.

Faza ponovne izbire

Ta faza ni nujna za delovanje, se pa uporabi v primeru, ko iniciator dokonča operacijo, katere ciljna naprava iz kakršnegakoli vzroka ni uspešno končala. V to fazo se vstopi tudi, če faza izbire ni uspela (ciljna naprava se ni odzvala v določenem času). V tem primeru lahko iniciator ponovno poizkuša izbrati isto napravo ali pa javi napako in sprosti vodilo.

2.1.3.2. Faze prenosa podatkov

Ko sta izbrana iniciator in naprava, s katero iniciator komunicira, se lahko prične prenos podatkov.

Prenos podatkov ima štiri faze, ki so:

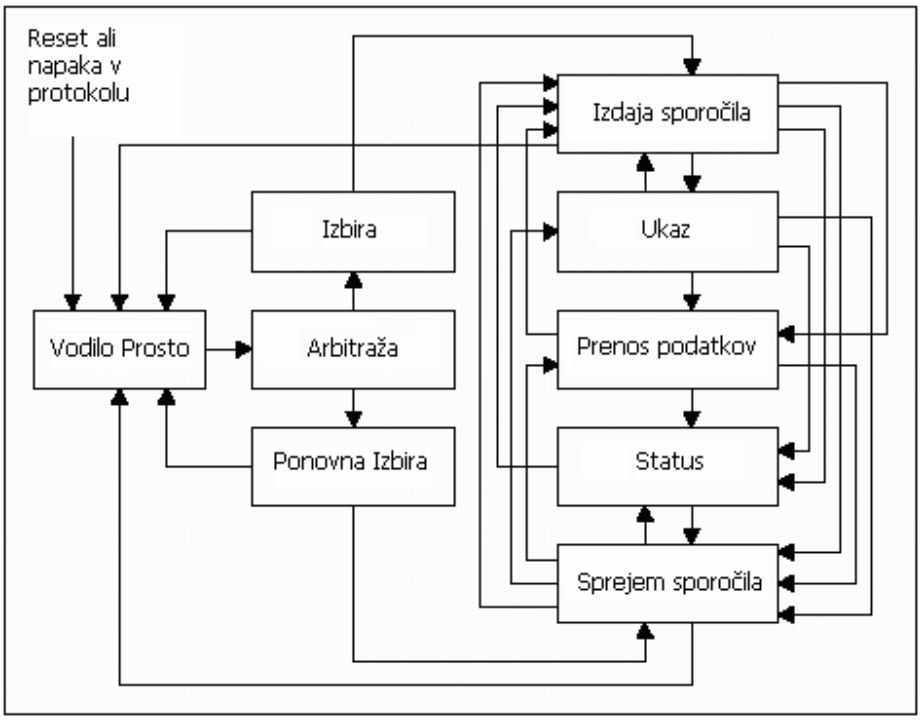
- izstava ukaza:
  v tej fazi pošlje inciator ukaz ciljni napravi.;
- prenos podatkov:
  prenos podatkov lahko poteka k ciljni naprave ali obratno. Za prenos podatkov sta praktično gledano definirani dve fazi, in sicer faza prenosa v ciljno napravo in faza prenosa iz ciljne naprave. V katero smer se podatki prenašajo, je določeno ob izdaji ukaza;
- status prenosa:

ciljna naprava sporoči iniciatorju o uspelem ali neuspelem prenosu podatkov. Iniciator se na to odzove s fazo sporočila;

- faza sporočila:

tudi tukaj sta definirani dve fazi, ki se ločita glede na smer pretoka podatkov. V tej fazi se lahko izmenja eno ali več sporočil, vendar vsa potekajo v enako smer;

Pri fazi prenosa podatkov se lahko prenaša informacija na asinhroni ali sinhroni način. Slednji je izbirni in ga ni treba implementirati. Podprt je lahko tudi širokopasovni prenos širine 16 ali 32 bitov. Nujno je za delovanje potreben samo osnovni 8-bitni prenos podatkov. Diagram prehajanja stanj na vodilu je prikazan na sliki 1.



Slika 1: Faze komuniciranja na vodilu SCSI

2.1.3.3. Izredna stanja na vodilu

Vodilo SCSI ima lahko dve izredni stanji, ki sta zahteva za pozornost (*attention*) in pa vrnitev v prvotno stanje (*reset*).

- Zahteva za pozornost

To stanje da iniciatorju možnost, da pošlje sporočilo ciljni napravi. Ciljna naprava preide v fazo prenosa sporočila.

- Reset

To stanje se uporabi za takojšnjo sprostitev vseh naprav z vodila SCSI. Stanje ima prednost pred vsemi ostalimi dogodki na vodilu in ga lahko vzpostavi katerakoli naprava, ki je priključena na vodilo.

Vpliv, ki ga ima vzpostavitev prvotnega stanja na vodilu na vhodno-izhodne procese, ki še niso bili dokončani, je odvisen od načina vodenja takšnega stanja s strani ciljne naprave. Taka načina sta lahko dva: tako imenovani trdi (*hard*) reset ali pa programski (*soft*) reset. Ta dva načina se med seboj izključujeta in vsaka ciljna naprava bi naj omogočala le enega od njiju.

Naprave, ki podpirajo trdi reset, bi se naj pri razpoznanem signalu reset odzvale tako, da bi prekinile vse vhodno-izhodne procese, sprostile vse rezervirane naprave SCSI, se vrnile v začetno stanje in si interno nastavile stanje pozornosti, nato pa se odzvale le na ukaze povpraševanja o stanju, ki bi ga na ukaz sporočile iniciatorju.

Naprave, ki podpirajo programski reset, bi naj pri razpoznanem signalu reset poizkusile končati vse vhodno-izhodne procese, ki so že bili določeni za izvedbo, ohranile naj bi vse rezervacije naprav SCSI, prav tako pa ostale v trenutnem stanju lastne rezervacije in ohranile trenutne vrednosti, ki so bile določene z ukazom MODE SELECT. Programski reset omogoča iniciatorju opraviti reset z minimalnimi motnjami ostalih iniciatorjev, ki so morda povezani na isto vodilo [3].

2.2. Ukazi

Ukazi SCSI, ki so nujni za implementacijo, so splošno definirani, tako da lahko podpirajo široko področje možnih naprav, ki bi bile priključene na vodilo. Število teh ukazov je zato zmanjšano na minimum. Ukazi se tako delijo na nujne, katere morajo podpirati vse naprave, izbirne, za katere se priporoča, da jih naprave podpirajo, specifične za proizvajalca, ki jih definira proizvajalec sam, in rezervirane, ki pa so predvideni za nadaljnje nadgradnje.

2.2.1. Sestava ukaza

Ukaz je zaporedje zlogov. Predvideni so ukazi dolžine 6, 10 in 12 zlogov (slike 2, 3 in 4). Prva beseda ukaza naj bi zmeraj vsebovala kodo ukaza, kot je definirana v mednarodnem standardu. Ta koda določi dolžino ukaza. Biti v ukazu, ki so določeni kot rezervirani, bi naj zmeraj imeli vrednost 0. V kolikor ta vrednost ni nič, lahko naprava ukaz zavrne z odgovorom, da je ukaz neveljaven.

<table>
<tr><th>Bit
Byte</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td colspan="8">Koda operacije</td></tr>
<tr><td>1</td><td colspan="3">Logična št. enote</td><td colspan="5">(MSB)</td></tr>
<tr><td>2</td><td colspan="8" rowspan="2">Logični naslov bloka (po potrebi)</td></tr>
<tr><td>3</td></tr>
<tr><td>4</td><td colspan="8">Dolžina prenosa podatkov (po potrebi)
Dolžina dodatnega prenosa (po potrebi)
Dolžina rezerviranega pomnilnika (po potrebi)</td></tr>
<tr><td>5</td><td colspan="8">Kontrolni zlog</td></tr>
</table>

Slika 2: Šablona ukaza, sestavljenega iz šest zlogov

Bit / Byte	7	6	5	4	3	2	1	0
0	Koda operacije							
1	Logična št. enote			Rezervirano				
2	(MSB)							
3	Logični naslov bloka (po potrebi)							
4								
5								(LSB)
6	Rezervirano							
7	(MSB)	Dolžina prenosa podatkov (po potrebi) Dolžina dodatnega prenosa (po potrebi) Dolžina rezerviranega pomnilnika (po potrebi)						
8								(LSB)
9	Kontrolni zlog							

Slika 3: Šablona ukaza, sestavljenega iz deset zlogov

Bit / Byte	7	6	5	4	3	2	1	0
0	Koda operacije							
1	Logična št. enote			Rezervirano				
2	(MSB)							
3	Logični naslov bloka (po potrebi)							
4								
5								(LSB)
6	(MSB)							
7	Dolžina prenosa podatkov (po potrebi) Dolžina dodatnega prenosa (po potrebi) Dolžina rezerviranega pomnilnika (po potrebi)							
8								
9								(LSB)
10	Rezervirano							
11	Kontrolni zlog							

Slika 4: Šablona ukaza, sestavljenega iz dvanajst zlogov

Koda ukaza je sestavljena iz dveh delov. Biti od 7 do 5 določajo tip ukaza (dolžino), ostali biti (od 4 do 0) pa določajo ukaz sam. Logična številka enote (LUN - *Logical Unit Number*) je lahko določena s posebnim sporočilom ali pa je dodana v opisu ukaza. V kolikor je LUN že prej bila določena s sporočilom, potem se polje LUN v ukazu ne upošteva. Polje LUN v ukazu je za SCSI-II

neuporabno, ker je posebno sporočilo nujno za implementacijo in ga nekatere naprave že uporabljajo za lastno identifikacijo svojih pod-delov (npr. knjižnice Storagetek).

Logični naslov bloka v logičnih enotah ali znotraj particije določa začetni blok, od katerega dalje se bo pričel prenos podatkov, katerih dolžina je določena v polju dolžine prenosa. Prav tako določata dolžina parametrov ter velikost rezerviranega pomnilnika število podatkov, ki se bodo prenašali v fazi prenosa podatkov.

Kontrolna beseda na koncu ukaza ima dva pomembna kazalca, ta sta "flag" in "link". Le eden od teh dveh je lahko naenkrat določen. Bit "link" je določen za povezavo vhodno-izhodnih procesov preko več ukazov. Ta bit določa, da naj naprava ukaz poveže z naslednjim in se po prenosu podatkov znova postavi v fazo izstave ukaza s statusom, da je vmesni ukaz uspel. Bit "flag" se uporablja za prekinitev več zaporednih ukazov.

2.2.2. Status

Statusna beseda se pošlje iniciatorju v fazi prenosa. Naprava preide v fazo prenosa statusa po vsakem uspelem ali neuspelem ukazu, razen če je bil ukaz prekinjen zaradi dogodka, kot so nasilna prekinitev ukaza (ABORT), reset vodila, posebni ukaz za praznjenje čakalne vrste (CLEAR QUEUE) ali nepričakovani izklop. Razčlenitev statusnega niza je prikazana na sliki 5.

Biti statusnega zloga								Status
7	6	5	4	3	2	1	0	
R	R	0	0	0	0	0	R	GOOD
R	R	0	0	0	0	1	R	CHECK CONDITION
R	R	0	0	0	1	0	R	CONDITION MET
R	R	0	0	1	0	0	R	BUSY
R	R	0	1	0	0	0	R	INTERMEDIATE
R	R	0	1	0	1	0	R	INTERMEDIATE-CONDITION MET
R	R	0	1	1	0	0	R	RESERVATION CONFLICT
R	R	1	0	0	0	1	R	COMMAND TERMINATED
R	R	1	0	1	0	0	R	QUEUE FULL
Vse ostale kode								Rezervirano
Legenda: R = Rezerviran bit								

Slika 5: Statusni niz

Vrnjene vrednosti statusnega niza so naslednje:

- GOOD

Ukaz je bil uspešno izveden.

- CHECK CONDITION

Ta status pomeni, da je bila zadržana vrednost (*contingent allegiance*). Glej poglavje »Zadržana vrednost«.

- CONDITION MET

Ta status se vrne, kadar je določena operacija izvedena (v praksi zelo redko uporabljeno).

- BUSY

Pove, da naprava trenutno ne more sprejeti izdanega ukaza, ker je zasedena. Priporoča se, da iniciator poizkusi ponovno izdati ukaz kasneje.

- INTERMEDIATE

Ta status pove, da se je ukaz v seriji povezanih ukazov izvedel.

- INTERMEDIATE CONDITION MET

Operacija v seriji povezanih operacij je izvedena.

- RESERVATION CONFLICT

Status pove, da je naprava trenutno rezervirana za uporabo v drugačnem obsegu, kot ga zavzema določen ukaz. Priporoča se, da iniciator poizkusi ukaz ponovno izdati kasneje.

- COMMAND TERMINATED

Ta status se nastavi, kadar je operacija bila prekinjena na željo iniciatorja.

- QUEUE FULL

Ta status naj bo implementiran samo, če je implementirano čakanje v vrsti. Če ni vrste, potem se ta ne more napolniti.

2.2.3. Zadržana vrednost

Stanje zadržane vrednosti se pojavi, kadar je vrnjen status CHECK CONDITION ali COMMAND TERMINATED. Ciljna naprava ostane v tem stanju, dokler iniciator podatkov (*sense data*) ne prebere ali pa se stanje prekine s posebnim sporočilom.

Dokler je ciljna naprava v stanju zadržane vrednosti, se na ostale zahteve odzove s statusom BUSY ali pa z enakim statusom, kot je bil pri vzpostavitvi tega stanja. Večina naprav se teh navodil ne drži. Te naprave sprejmejo nove ukaze, podatke zadržane vrednosti prejšnjega ukaza pa izbrišejo oziroma jih prepišejo z zadržanimi vrednostmi novega ukaza, v kolikor je do njih prišlo. Posledično temu sledi, da lahko podatke zadržane vrednosti preberemo iz naprave le enkrat, kajti branje teh podatkov zahteva poseben ukaz (REQUEST SENSE), ki podatke potem, ko jih prebere, prepiše z rezultatom lastnega izvajanja.

2.2.3.1. Struktura zadržanih podatkov

Podatki zadržane vrednosti imajo določen format, kot je prikazan na sliki 6.

Bit / Byte	7	6	5	4	3	2	1	0
0	Valid	Error code (70h or 71h)						
1	Segment number							
2	Filemark	EOM	ILI	Reserved	Sense key			
3 – 6	(MSB) Information (LSB)							
7	Additional sense length (n-7)							
8 – 11	(MSB) Command-specific information (LSB)							
12	Additional sense code							
13	Additional sense code qualifier							
14	Field replaceable unit code							
15 – 17	SKSV	Sense-key specific						
18 – n	Additional sense bytes							

Slika 6: Podatki zadržane vrednosti

Bit "filemark" pove, ali je pri branju magnetnega traka prišlo do oznake na traku. V tem primeru ukaz READ vrne 0 in se ta bit postavi na 1.

Bit "EOM" (*end of media*) pove, da je prišel trak do konca.

Polje "sense key" lahko ima vrednosti od 0x00 do 0x0F (šestnajstiško) in ima eno izmed naslednjih vrednosti:

NO SENSE, RECOVERED ERROR, NOT READY, MEDIUM ERROR, HARDWARE ERROR, ILLEGAL REQUEST, UNIT ATTENTION, DATA PROTECT, BLANK CHECK, VENDOR SPECIFIC, COPY ABORTED, ABORTED COMMAND, EQUAL, VOLUME OVERFLOW, MISCOMPARE ali RESERVED.

ASC (*additional sense code*) in ASCQ (*additional sense code qualifier*) skupaj podata dodatne informacije o napaki. Uporabnik lahko iz tabele natančno ugotovi, do kakšne napake je prišlo, in se nato ustrezno odzove. Mnogo naprav ima definirane lastne dodatne tabele, kjer so opisane napake, ki so specifične za posamezne tipe naprav.

2.2.4. Izredno stanje na napravi (*unit attention*)

Ciljna naprava generira izredno stanje po vsakem sporočilu za reset, po trdem resetu vodila ter ob vklopu naprave. Prav tako se izredno stanje vzpostavi ob naslednjih dogodkih:

- izmenljivi medij se je zamenjal,
- notranji parametri so se spremenili na zahtevo drugega inciatorja,
- revizija programja v napravi se je spremenila ali
- zgodili so se drugi dogodki, ki zahtevajo izredno stanje.

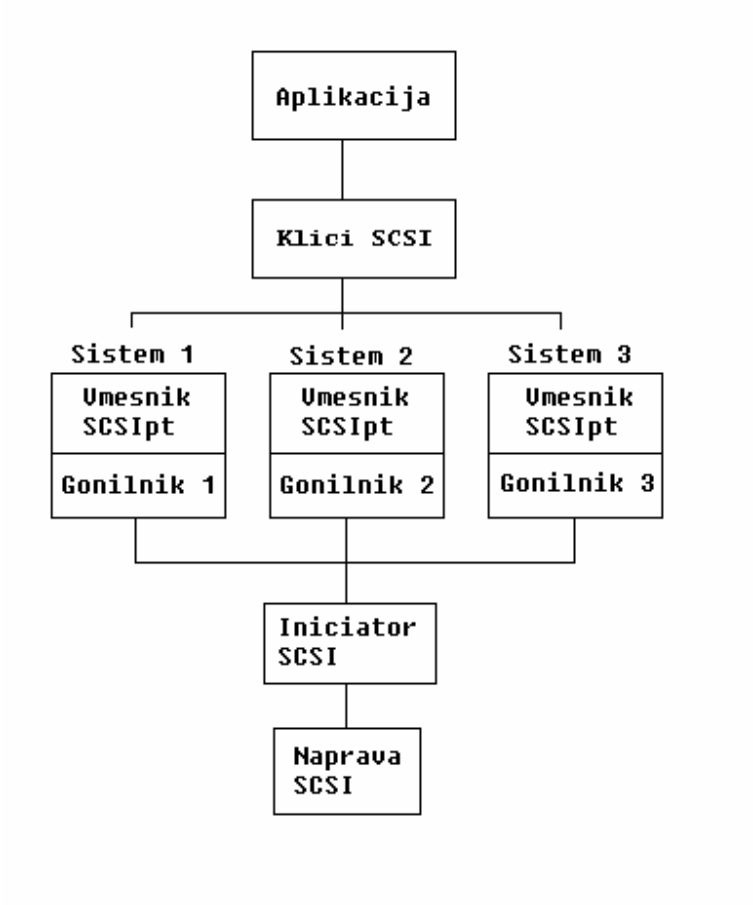
Izredno stanje se konča, ko iniciator prebere zadržane podatke v napravi. Drugi ukazi se ne izvedejo in vrnejo status CHECK CONDITION. Izjema je le ukaz INQUIRY, ki pod določenimi pogoji lahko zbriše izredno stanje v ciljni napravi (npr. pri spremembi notranjih parametrov) [3].

3. Problem sistemske neodvisnosti

Ko govorimo o standardu SCSI, se pri tem ne določi operacijski sistem, pod katerim bi se ta naj izvajal. Prav nasprotno! Osnovna ideja standarda je, da je sistemsko neodvisen. Tako je implementacija tega v rokah proizvajalca operacijskega sistema. Zato je le-ta implementiran na vsakem sistemu drugače. Nekako v navado je prišlo, da se ta standard že implementira v sistem kot priložen gonilnik, vendar pa ni prisoten pri vseh sistemih, zato za določene sisteme obstajajo gonilniki, ki so jih napisali drugi proizvajalci in jih je treba na sistem dodatno namestiti. Ti gonilniki omogočajo programu dostop do priključenih naprav SCSI ter definirajo postopke za pošiljanje ukazov tem napravam. Med seboj se razlikujejo, ponekod so razlike malenkostne, ponekod pa se razlikuje celoten postopek dostopa.

Za proizvajalca programske opreme je lahko neugodno, v kolikor želi svoj izdelek, ki deluje z neko napravo na določenem sistemu, prenesti na sistem drugega proizvajalca. Tako se je pojavil problem pri programski opremi HP Openview Data Protector, kjer smo želeli, da bi se modul za dostop do naprav SCSI lahko izvajal oz. prevedel pod več sistemi. Ideja je napisati programski vmesnik, ki bi te arhitekturne razlike med gonilniki raznih operacijskih sistemov naredil transparentne za programske dele višjega nivoja. Tako je nastal projekt zasnove sistemsko neodvisnega modula SCSI, imenovan SCSIpt (*SCSI pass through*).

Zasnova modula je omogočiti klicem višjega nivoja sistemsko neodvisnost (slika 7).



Slika 7: Sistemska odvisnost modulov

3.1. Arhitektura SCSI-implementacije

Kot vodilo za arhitekturo so bili definirani naslednji pogoji:

- modul mora biti dovolj splošen, da se lahko uporabi za mešane zasnove sistemov;
- biti mora neodvisen od višjih nivojev, kot vhod sprejme samo opisni parameter datoteke;
- definira API (*application interface*) za določene ukaze SCSI;
- pusti možnost dodajanja novih ukazov SCSI oz. definira splošni API-klic za univerzalni ukaz SCSI;
- sistemsko specifične funkcije se izvedejo na čim nižjem nivoju v modulu, medtem ko se odločitve sprejemajo neodvisno od sistema, na katerem se izvajajo.

Definirali smo univerzalno strukturo SCSI, ki je uporabljena za klice SCSI in jo uporablja SCSIpt API:

typedef struct ScsiPtTag

```
{
      UCHAR cdb[16];          /* opis ukaza (največ 16 zlogov) */
      UINT  cdblength;        /* dolžina ukaza */
      UCHAR *data;            /* kazalec na podatke */
      ULONG datalength;       /* dolžina podatkov */
      UCHAR datadirection;    /* smer prenosa podatkov */
      UINT  timeout;          /* čas čakanja, da se ukaz izvede */
      UCHAR *sense;           /* podatki zadržane vrednosti */
      ULONG senselength;      /* dolžina podatkov zadržane vrednosti */
      UCHAR returnstatus;     /* statusni zlog po izvedbi ukaza */
} ScsiPt;
```

3.2. Programska struktura

Definirani so naslednji programski nivoji oz. funkcije, ki se kličejo zaporedno ena za drugo.

SCSIPT_ScsiCommand()
SCSIPT_io()
SCSIPT_command()
SCSIPT_DoIO()
SCSIPT_ioctl()

3.2.1. SCSIPT_ScsiCommand()

Ta funkcija je funkcija za univerzalni klic SCSI, njeni parametri se neposredno preslikajo v strukturo Scsipt in lahko vsebuje kakršenkoli ukaz.

3.2.2. SCSIPT_io()

Ker se lahko vsak ukaz SCSI izvede neuspešno, se naprava SCSI postavi v stanje zadržane vrednosti in v internem pomnilniku se kreirajo podatki o tem stanju. Nekateri gonilniki vračajo te podatke avtomatsko ali pa to možnost dopuščajo.

SCSIpt API uporablja to možnost in vrne podatke skupaj s stanjem vsakokrat, tudi če gonilnik tega ne podpira, zato se tukaj rezervira pomnilnik za te podatke.

Nekateri ukazi ne uspejo, ker se je v napravi zgodil kakšen dogodek in želi to naprava sporočiti ne glede na to, kakšen ukaz je bil za napravo izdan. Ker večinoma ti podatki ne vplivajo direktno na izdani ukaz, se lahko opustijo in se ukaz izvede ne glede na vzpostavljeno stanje, zato se v tej funkciji dopušča možnost večkratnega klica istega ukaza, če prejšnji klic ni uspel. Obstajata dve možnosti ponavljanja ukaza. Ena možnost je, da se ukaz ponovi takoj, pri drugi možnosti pa se ukaz ponovi po določenem času. Prav tako je možna izbira, kolikokrat se ukaz ponovi, preden se vrne sporočilo o napaki. Ta dva parametra se podata v obliki globalne spremenljivke (*environment variable*). Vrednosti so shranjene v spremenljivkah SCSIPT_RETRYCOMMAND in SCSIPT_SLEEPCOMMAND. Slednja določa čas v sekundah, ko program čaka na ponoven poskus izdaje ukaza.

Tukaj se prav tako preverijo osnovne vrednosti strukture ScsiPt, če le ta vsebuje vrednosti, ki so nujne za delovanje (npr. veljaven opis ukaza). Preveri se čas trajanja poskusa izvedbe ukaza za gonilnik (*timeout*) in se nastavi na privzeto vrednost, v kolikor ta ni predhodno vnesen.

Na koncu funkcije se v zanki kliče SCSIPT_command(). Zanka se prekine, če se funkcija uspešno konča ali če sporoči kritično napako, zaradi katere se ponoven poskus klica ne obnese.

3.2.3. SCSIPT_command()

Včasih se ukaz ne izvede, ker naprava, s katero želimo delati, trenutno izvaja kakšen drug ukaz ali je kakorkoli drugače zasedena. V tem primeru gonilnik vrne status, s katerim pove, da naprava trenutno ne more izvršiti ukaza. Zato je v tej funkciji vnesena nova zanka, ki ponavlja ukaz, dokler ni naprava znova prosta oziroma dokler se ne izvede določeno število poskusov. Tudi ti se določajo z globalnima spremenljivkama. Ti spremenljivki sta OB2SPTRETRY in OB2SPTSLEEP, od katerih slednja določa čakalno dobo v sekundah. Zanki, ki nastaneta zaradi notranjega stanja naprave in zaradi statusa, ki ga vrača gonilnik, sta ločeni zato, da se lahko izvede natančnejša kontrola nad ponovnimi preverjanji ukazov glede na napako, ki se je zgodila.

V kolikor klic funkcije SCSIPT_DoIO() uspe, SCSIPT_command() signalizira uspeh, v nasprotnem primeru pa preveri, ali je vrnjen status zadržane vrednosti. V kolikor podatki o stanju niso vrnjeni v prej rezerviranem pomnilniku, se prevzamejo iz naprave s klicem ukaza REQUEST SENSE. Ker se ta ukaz kliče rekurzivno z isto API-funkcijo, je tukaj treba preveriti, ali je trenutni

klic že bil taisti REQUEST SENSE, kajti v tem primeru so podatki o stanju že v podatkovnem pomnilniku in jih ni treba ponovno zahtevati.

Naprava je lahko zavrnila ukaz, ker je bila trenutno zasedena, pa je gonilnik to spregledal, zato tukaj preverimo, ali podatki o zadržani vrednosti identificirajo napravo kot zasedeno. V tem primeru se status o ponavljanju ukaza prenese nivo višje in se uporabi ponoven klic iz funkcije SCSIPT_io().

V tem delu tudi reagiramo na status izrednega stanja, in sicer tako, da se izredno stanje prekliče in se ukaz ponovi.

Če naprava vrne kakršenkoli drugačen status, funkcija vrne napako in pove, da so podatki zadržane vrednosti na voljo za dodatno analizo, ki se lahko izvede na višjem nivoju.

3.2.4. SCSIPT_DoIO()

V tej funkciji se izpolnijo strukture, ki so specifične za določen gonilnik na določenem sistemu. Interna struktura ScsiPt se preslika v strukturo, ki si jo je definiral gonilnik za določen sistem.

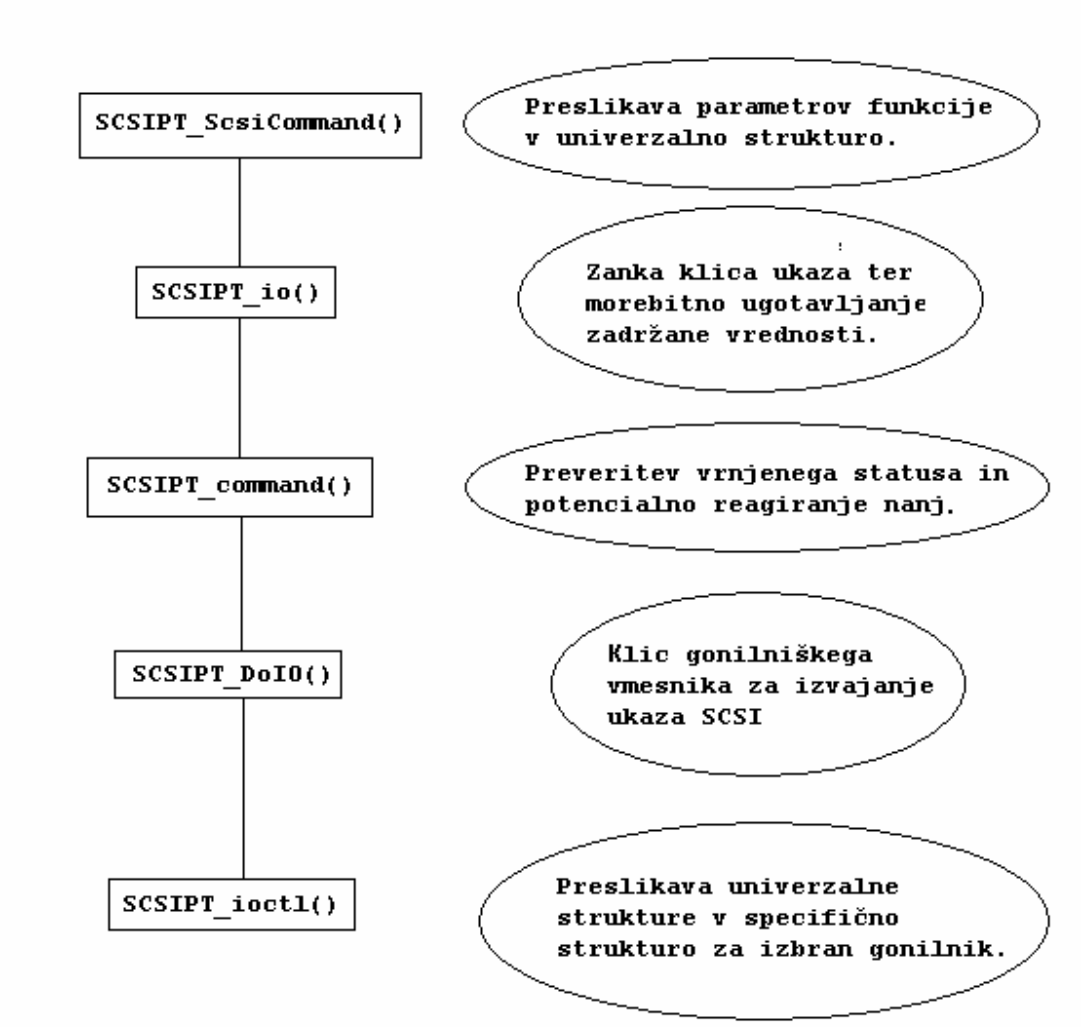
Funkcija preveri smer prenosa podatkov. V kolikor se podatki prenašajo v smeri iz naprave, se počisti pomnilnik za sprejem podatkov (prepiše se z ničlami).

Nato se pokliče funkcija SCSIPT_ioctl(), ki izvede sistemski klic. V kolikor klic uspe, se popravijo izhodni podatki v strukturi ScsiPt. Ti so vrnjeni status ter dolžina prenosa podatkov. V kolikor je prišlo do prenosa podatkov zadržane vrednosti, se tudi ti podatki prepišejo v prej rezerviran pomnilnik za te podatke. Prav tako se ažurira dolžina podatkov, prenesenih iz zadržane vrednosti.

3.2.5. SCSIPT_ioctl()

Ta funkcija izvede sistemski klic za določen sistem. Vsak ima svoje posebne lastnosti, ki so natančneje opisane v poglavju o sistemih. V splošnem se tukaj preveri uspeh izvedenega klica in ali je funkcija bila neuspešno izvedena zaradi tega, ker je bila ciljna naprava zasedena.

Programska struktura je okvirno prikazana na sliki 8.



Slika 8: Programska struktura

3.3. Programske reference SCSIpt modula in SCSIpt API funkcij

3.3.1. SCSIPT_io()

Funkcija SCSIPT_io izvede operacijo SCSI.

int SCSIPT_io (

```
int         fd,         /* opis zbirke */
ScsiPt      *scsiPt     /* kazalec na strukturo ScsiPt */
);
```

Parametri:

fd

opisni parameter za podatkovno zbirko;

scsiPt

kazalec na strukturo ScsiPt, ki vsebuje podatke za izvedbo ukaza SCSI.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

Opombe:

Ta funkcija se kliče neposredno iz API-funkcij SCSIpt. V kolikor programer sam kreira in zapolni strukturo ScsiPt z veljavnimi vrednostmi, lahko pokliče to funkcijo neposredno iz svoje aplikacije, vendar se priporoča univerzalni klic SCSIPT_ScsiCommand() iz API-selekcije SCSIpt.

3.3.2. SCSIPT_ScsiCommand()

Funkcija SCSIPT_ScsiCommand predstavlja splošen klic za SCSIpt API. S tem klicem lahko programer implementira ukaze, ki so specifičnega pomena za določenega proizvajalca oziroma produkt.

```
int
SCSIPT_ScsiCommand(
      tchar *fName,           /* ime kličoče funkcije */
      int fd,                 /* opis zbirke */
      UCHAR cdb[16],          /* podatki o ukazu SCSI */
      UCHAR cdblength,        /* dolžina ukaza */
      UCHAR *data,            /* kazalec na pomnilnik s podatki */
      ULONG datalength,       /* dolžina prenosa podatkov */
      UCHAR datadirection,    /* smer prenosa podatkov */
      ULONG timeout,          /* čas, ki ga sistemski klic čaka, */
                              /* da se operacija izvede */
      UCHAR *sense,           /* kazalec na pomnilnik z zadržanimi podatki */
      ULONG senselength,      /* dolžina pomnilnika za zadržane podatke */
      ULONG *returned         /* dejanska količina prenesenih podatkov */
);
```

Parametri:

fName

Ime kličoče funkcije, lahko NULL.

fd

Opisni parameter za podatkovno zbirko.

cdb

Te vrednosti se neposredno preslikajo v ukaz SCSI. Ta izbira omogoča neodvisnost od dejanskih podatkih samega ukaza glede na izvedeno operacijo.

cdblength

Dolžina ukaza, lahko 6, 10, 12 ali 16 zlogov. Dolžine 16 nekateri gonilniki ne podpirajo, v tem primeru se klic zavrne.

data

Kazalec na pomnilnik, ki vsebuje podatke, ki se prenašajo bodisi iz naprave ali v njo. Če se podatki ne prenašajo, potem se ta parameter poda kot NULL, parameter »datalength« pa se postavi na vrednost 0.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

datadirection

Smer prenosa podatkov. Dovoljene vrednosti so:

SCSIPT_DATA_READ - podatki se prenašajo v smeri iz naprave;

SCSIPT_DATA_WRITE - podatki se prenašajo v napravo;

SCSIPT_DATA_NONE - prenosa podatkov ni. V tem primeru morata biti parametra data in datalength nastavljena na vrednosti NULL in 0.

timeout

Čas, izražen v sekundah, ki ga bo sistemski klic porabil za čakanje na izvedbo ukaza. V kolikor se ukaz ne izvede v tem času, bo sistemski gonilnik javil napako.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo. Prav tako mora imeti v tem primeru parameter »senselength« vrednost 0.

senselength

Velikost rezerviranega pomnilnika za podatke zadržane vrednosti.

returned

Izhodni podatek. V kolikor je prišlo do prenosa podatkov, vsebuje ta spremenljivka dolžino, ki se je dejansko prenesla. Ta vrednost ni nujno enaka vrednosti v »datalength«.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.3. SCSIPT_Inquiry()

Funkcija vrne informacijo o napravi ali o napravah, ki so nanjo priključene, dodatni parameter omogoča vrnitev podrobnejših informacij o napravi.

```
int
SCSIPT_Inquiry (
      int fd,                 /* opis zbirke */
      UCHAR evpd,             /* vklop vitalnih podatkov o produktu */
      UCHAR page,             /* zahteva po dodatni informaciji */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      UCHAR datalen,          /* dolžina prenosa podatkov */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

evpd

Vklopi vrnitev podrobnejših informacij o produktu. V kolikor ta parameter ni enak nič, mora parameter »page« vsebovati indeks strani o dodatnih podatkih o produktu. V kolikor je evpd enak 0, potem vrnjeni podatki vsebujejo standardni format informacije o produktu.

page

V kolikor želimo izvedeti podrobnejše informacije o produktu, potem lahko zahtevamo posebne informacije pri tem ukazu. Nekatere strani so opisane v standardu SCSI, druge pa so lahko specifične za določen produkt. V kolikor želimo dobiti posebne informacije o produktu, potem tukaj določimo indeks strani o posebnih podatkih. Če določimo parametru »page« vrednost 0, potem se v podatkih vrne standardna informacija o produktu.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.4. SCSIPT_RequestSense()

Funkcija vrne podatke o zadržani vrednosti.

```
int
SCSIPT_RequestSense (
      int fd,                 /* opis zbirke */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      UCHAR datalength        /* dolžina prenosa podatkov */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

Datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.5. SCSIPT_TestUnitReady()

Ukaz preveri, ali je naprava pripravljena na delovanje. Ta ukaz ne sproži ponovnega testiranja naprave, temveč samo preveri, ali je naprava pripravljena na delovanje z medijem, ne da bi pri takšni operaciji prišlo do zadržane vrednosti.

```
int
SCSIPT_TestUnitReady (
      int fd,                 /* opis zbirke */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.6. SCSIPT_Reserve()

Naprava se rezervira za ekskluzivno uporabo iniciatorju. Medtem ko je naprava rezervirana za določenega iniciatorja, vrne napako vsakemu ukazu, ki izvira iz katerekoli druge naprave razen tiste, ki si je ciljno napravo rezervirala.

```
int
SCSIPT_Reserve (
      int fd,                 /* opis zbirke */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.7. SCSIPT_Release()

Ukaz odklene napravo za uporabo drugim iniciatorjem. Iniciator odklene ciljno napravo takrat, ko ne potrebuje več ekskluzivnega dostopa ali pa je s svojimi operacijami končal.

```
int
SCSIPT_Release (
      int fd,                 /* opis zbirke */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.8. SCSIPT_ModeSense()

Ta ukaz omogoča, da se vrnejo trenutno nastavljeni parametri v napravi sami. Nasprotno se lahko parametri v napravi nastavijo z ukazom ModeSelect.

```
int
SCSIPT_ModeSense (
      int fd,                 /* opis zbirke */
      UCHAR page,             /* indeks strani s parametri */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      UCHAR datalength,       /* dolžina prenosa podatkov */
      int dbd,                /* prepreči vrnitev opisa blokov */
      ULONG *returned,        /* velikost vrnjenih podatkov */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

Page

Parameter je sestavljen iz dveh delov. Prvi del sta prva dva bita, ki določata tip podatkov, ki jih bo ukaz vrnil. Tipi podatkov so lahko trenutne vrednosti, spremenljive vrednosti, privzete vrednosti ali shranjene vrednosti. Drugi del pa vsebuje indeks strani, ki jih bo ukaz vrnil. Razne strani vsebujejo različne podatke o vrednostih v napravi.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

dbd

Vrednost 0 določa, da vrnjeni podatki lahko vsebujejo opise blokov, medtem ko vrednost ena določa, da teh opisov vrnjeni podatki ne vsebujejo.

returned

Izhodni podatek. V kolikor je prišlo do prenosa podatkov, vsebuje ta spremenljivka dolžino, ki se je dejansko prenesla. Ta vrednost ni nujno enaka vrednosti v »datalength«.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.9. SCSIPT_ModeSelect()

```
int
SCSIPT_ModeSelect (
      int fd,                 /* opis zbirke */
      int pf,                 /* »page format«: 0=scsi1; 1=scsi2 */
      int sp,                 /* »save page«; 0=trenutna sprememba */
                              /* 1=shrani v trajen spomin */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      UCHAR datalength,       /* dolžina prenosa podatkov */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
)
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

pf

Parameter določa format zapisa strani, in sicer vrednost 0 določa SCSI-1 in vrednost 1 določa SCSI-2.

sp

Določi, ali se naj vrednosti, poslane v napravo, shranijo v trajni pomnilnik. Vrednost 0 določa, da naj se podatki ne shranijo v trajni pomnilnik in se tako izgubijo ob možnem izpadu energije ali pri stanju reset na vodilu.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec data.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.10. SCSIPT_LogSense()

Ukaz omogoča, da se iz naprave prebere informacija o statistiki delovanja naprave ali njenih podenot.

```
int
SCSIPT_LogSense (
      int fd,                 /* opis zbirke */
      int pc,                 /* nadzor strani (page control) */
      UCHAR page,             /* indeks strani */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      ULONG datalength,       /* dolžina prenosa podatkov */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

pc

Ta parameter pove vrsto podatkov, ki se naj vrnejo na zahtevani strani podatkov.

To so lahko npr. samo podatki, ki se dajo spremeniti, ali pa samo podatki, ki so bili nazadnje spremenjeni ipd.

page

Vsebuje indeks strani, ki jih bo ukaz vrnil. Razne strani vsebujejo različne podatke o statističnih vrednostih v napravi.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.11. SCSIPT_SendDiagnostic()

Ta ukaz zahteva od naprave, da opravi diagnostiko svojega delovanja ali kakšne druge enote. Naprava opravi samotestiranje svojih sposobnosti in s tem preveri lastno sposobnost za delovanje.

```
int
SCSIPT_SendDiagnostic (
      int fd,                 /* opis zbirke */
      UCHAR flags,            /* dodatni parametri */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      ULONG datalength,       /* dolžina prenosa podatkov */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

fd

Opisni parameter za podatkovno zbirko.

flags

Parameter določa dodatne parametre k zahtevi. Ti so:

UnitOfl (bit 0): 1=dovoli uporabo medija za samopreverjanje,
DevOfl (bit 1): 1=dovoli uporabo podenot za samopreverjanje,
SelfTest (bit 2): 1=samopreverjanje; 0=upoštevaj parametre,
PageFmt (bit 4): format strani, 1=standardna.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.12. SCSIPT_Direct_ReadCapacity()

Ukaz vrne podatke o kapaciteti naprave, ki deluje z direktnim dostopom do medija.

```
int
SCSIPT_Direct_ReadCapacity (
      int fd,                 /* opis zbirke */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.13. SCSIPT_Direct_Read10()

Ukaz prebere podatke z medija, ki ima direkten dostop (disk).

```
int
SCSIPT_Direct_Read10 (
      int fd,                 /* opis zbirke */
      ULONG blockPos,         /* položaj prvega bloka zapisa */
      ULONG blockNum,         /* število prebranih blokov */
      ULONG blockSize,        /* velikost enega bloka */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

blockPos

Parameter poda položaj prvega bloka na mediju, s katerega se bodo začeli brati podatki.

blockNum

Parameter pove število blokov, ki se bodo prebrali z medija.

blockSize

Parameter poda velikost enega bloka.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.14. SCSIPT_Direct_Write10()

Ukaz zapiše podatke na medij.

```
int
SCSIPT_Direct_Read10 (
      int fd,                 /* opis zbirke */
      ULONG blockPos,         /* položaj prvega bloka zapisa */
      ULONG blockNum,         /* število prebranih blokov */
      ULONG blockSize,        /* velikost enega bloka */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

blockPos

Parameter poda položaj prvega bloka na mediju, na katerega se bodo začeli zapisovati podatki.

blockNum

Parameter pove število blokov, ki se bodo zapisali na medij.

blockSize

Parameter poda velikost enega bloka.

data

Kazalec na pomnilnik, v katerega se bodo zapisali podatki.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.15. SCSIPT_Seq_Write6()

Ukaz zapiše podatke na medij, ki ima zaporeden dostop (trak).

```
int
SCSIPT_Seq_Write6 (
      int fd,                 /* opis zbirke */
      UCHAR fixed,            /* tip zapisa blokov */
      ULONG size,             /* dolžina zapisa */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      ULONG datalength,       /* dolžina prenosa podatkov */
      ULONG *returned,        /* velikost dejansko prebranih podatkov */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

fixed

Pove, ali velikost zapisa poda dolžino v zlogih ali pa poda število blokov. V kolikor ima naprava na izbiro samo eno velikost bloka, potem mora ta parameter biti 1, drugače pa 0.

size

Poda velikost zapisa: v primeru, ko je »fixed« enak 1, ta parameter poda število blokov, ki se bodo zapisali na medij, v kolikor pa je »fixed« enak 0, ta parameter poda število zapisanih zlogov in mora biti enak »datalength«.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

Returned

Pove količino dejansko prebranih podatkov, ki ni nujno enaka želeni.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.16. SCSIPT_Seq_Read6()

Ukaz prebere podatke z medija, ki ima zaporeden dostop (trak).

```
int
SCSIPT_Seq_Read6 (
      int fd,                 /* opis zbirke */
      UCHAR fixed,            /* tip zapisa blokov */
      ULONG size,             /* dolžina zapisa */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      ULONG datalength,       /* dolžina prenosa podatkov */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

fixed

Pove, ali velikost zapisa poda dolžino v zlogih ali pa poda število blokov. V kolikor ima naprava na izbiro samo eno velikost bloka, potem mora ta parameter biti 1, drugače pa 0.

size

Poda velikost zapisa: v primeru, ko je »fixed« enak 1, ta parameter poda število blokov, ki se bodo prebrali z medija, v kolikor pa je »fixed« enak 0, ta parameter poda število prebranih zlogov in mora biti enak »datalength«.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec »data«.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.17. SCSIPT_Space()

Klic ponuja vrsto možnosti nastavljanja pozicije pri napravah z zaporednim prenosom podatkov. Parameter »code« določa tip ter način nastavitve medija na želeno pozicijo.

```
int
SCSIPT_Space (
      int fd,                 /* opis zbirke */
      UCHAR       code,       /* način skoka */
      ULONG       count,      /* smer in dolžina skoka */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

parametri:

fd

Opisni parameter za podatkovno zbirko.

code

Parameter pove tip skoka na mediju. Možne vrednosti so:

0 = bloki,
1 = zbirne oznake (*filemarks*),
2 = zaporedne zbirne oznake,
3 = nastavitev na konec zapisa,
4 = oznake za gručo (*setmarks*),
5 = zaporedne oznake za gručo.

count

Pove, koliko določenih tipov se naj preskoči. Vrednost je lahko tudi negativna, če se skok izvaja v smeri previjanja nazaj.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.18. SCSIPT_Rewind()

Ukaz previje medij na začetek trenutne particije. Pred tem ukazom je treba poskrbeti, da se vsi podatki, ki še niso bili zapisani na trak, nanj zapišejo.

```
int
SCSIPT_Rewind (
      int fd,                 /* opis zbirke */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.19. SCSIPT_LoadUnload()

S tem ukazom je mogoče medij naložiti v enoto ali izvreči iz nje. Večina današnjih enot podpira le možnost, da medij izvržejo, medtem ko se za nalaganje uporablja drug ukaz (*move*).

```
int
SCSIPT_LoadUnload (
      int fd,                 /* opis zbirke */
      UCHAR load,             /* 1=naloži; 0=izvrzi */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

load

Določa, ali se bo medij naložil v napravo ali pa bo izvržen iz nje. V kolikor ima »load« vrednost 1, potem se medij naloži v enoto in se postavi na začetek prve particije, v kolikor pa je vrednost 0, se medij izvrže iz enote.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.20. SCSIPT_WriteFilemarks()

Ukaz zapiše določeno število zbirnih oznak na medij.

```
int
SCSIPT_WriteFilemarks (
      int fd,                 /* opis zbirke */
      ULONG count,            /* število zapisanih zbirnih oznak */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

count

Poda število zapisanih zbirnih oznak.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.21. SCSIPT_Erase()

Ta ukaz povzroči, da se s celotnega ali z dela medija izbrišejo vsi podatki.

```
int
SCSIPT_Erase (
    int fd,
    sptSense *sense
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.22. SCSIPT_ReadBlockLimits()

S tem ukazom naprava sporoči velikosti blokov, ki jih medij premore. Vrnjeni podatki določajo najmanjšo, največjo ter vmesne možne velikosti enega bloka.

```
int
SCSIPT_ReadBlockLimits (
      int fd,                 /* opis zbirke */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.23. SCSIPT_MediumRemoval()

Ukaz prepreči ali dovoli odstranitev medija. Ukaz je neodvisen od stanja Reserve/Release.

```
int
SCSIPT_MediumRemoval (
      int fd,                 /* opis zbirke */
      UCHAR prevent,          /* 1 = prepreči; 0 = dovoli */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

prevent

Pove, ali se odstranitev medija dovoli ali prepreči. Vrednost 1 odstranitev medija prepreči, medtem ko jo vrednost 0 dovoli.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.24. SCSIPT_TapeAlert()

Razširjena funkcija, ki vrne dodatne vrednosti o napaki na traku. Ta ukaz ni del standarda SCSI, temveč je bil dogovorjen med proizvajalci opreme.

```
int
SCSIPT_TapeAlert (
      int fd,                 /* opis zbirke */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      ULONG datalength,       /* dolžina prenosa podatkov */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalen

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec data.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.25. SCSIPT_ReadElementStatus

Ukaz vrne status notranjih elementov v napravi.

```
int
SCSIPT_ReadElementStatus (
      int fd,                 /* opis zbirke */
      UCHAR volTag,           /* zahteva po vrnitvi labele elementa */
      UCHAR elType,           /* tip elementa */
      ULONG elStart,          /* začetni element */
      ULONG elNum,            /* število elementov */
      UCHAR *data,            /* kazalec na rezerviran pomnilnik za podatke */
      ULONG datalen,          /* dolžina prenosa podatkov */
      UCHAR IDcontrol,        /* zahteva po identifikaciji elementa */
      UCHAR control,          /* dodatna kontrola klica */
      ULONG timeout,          /* čas, ko sistemski klic čaka, */
                              /* da se operacija  izvede */
      ULONG *returned,        /* količina dejansko prenesenih podatkov */
      sptSense    *sense      /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

volTag

Ta parameter določa, da se naj skupaj s podatki o statusu notranjega elementa sporoči tudi labela elementa, v kolikor je ta funkcija podprta.

elType

Pove tip elementa, o katerem naj naprava poroča. Tipi so:

0 - vsi tipi,
1 - element za transport medija,
2 - element za shranjevanje medijev,
3 - element za importiranje ali eksportiranje,
4 - element za prenos podatkov.

elStart

Določa, od katerega elementa naprej se naj kreira poročilo. V poročilu bodo sporočeni samo elementi, katerih tip je enak elType in katerih indeks ni manjši od ElStart.

elNum

Določa število vrnjenih elementov.

data

Kazalec na pomnilnik, v katerem bodo vrnjeni podatki.

datalength

Dolžina prenosa podatkov. Ta vrednost mora odražati velikost rezerviranega pomnilnika, na katerega kaže kazalec data.

idControl

Parameter pove, da se naj skupaj s podatki o elementu vrne tudi njegova identifikacija.

control

Dodatna kontrola klica - nekatere knjižnice potrebujejo ta parameter za pravilno vračanje podatkov.

timeout

Čas, izražen v sekundah, ki ga bo sistemski klic porabil za čakanje na izvedbo ukaza. V kolikor se ukaz ne izvede v tem času, bo sistemski gonilnik javil napako.

returned

Izhodni podatek. V kolikor je prišlo do prenosa podatkov, vsebuje ta spremenljivka dolžino, ki se je dejansko prenesla. Ta vrednost ni nujno enaka vrednosti v datalength.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

3.3.26. SCSIPT_MoveMedium()

Ukaz premakne medij iz elementa v element.

```
int
SCSIPT_MoveMedium (
      int fd,                 /* opis zbirke */
      ULONG tea,              /* naslov transportnega elementa */
      ULONG sea,              /* naslov izvornega elementa */
      ULONG dea,              /* naslov ponornega elementa */
      UCHAR invert,           /* rotacija za 90 stopinj */
      ULONG timeout,          /* čas, ki ga sistemski klic čaka, */
                              /* da se operacija izvede */
      sptSense *sense         /* kazalec na pomnilnik z zadržanimi podatki */
);
```

Parametri:

fd

Opisni parameter za podatkovno zbirko.

tea

Indeks transportnega elementa. Če je ta vrednost enaka 0, potem se vzame privzeti transportni element, v kolikor to možnost naprava podpira.

sea

Indeks izvornega elementa. Od tukaj se bo vzel medij in se prenesel v ponor.

dea

Indeks ponornega elementa. Medij bo prenesen v ta element.

invert

Pove, ali se naj medij vstavi v ponorni element invertirano. Ta vrednost pride v poštev pri dvostranskih diskih.

timeout

Čas, izražen v sekundah, ki ga bo sistemski klic porabil za čakanje na izvedbo ukaza. V kolikor se ukaz ne izvede v tem času, bo sistemski gonilnik javil napako.

sense

Rezerviran pomnilnik za podatke zadržane vrednosti. Vrednost je lahko NULL, vendar se v tem primeru podatki o zadržani vrednosti ne vrnejo.

Vrnjena vrednost:

SCSIPT_RETURN_OK - ukaz se je izvedel uspešno;

SCSIPT_RETURN_CHECK_CONDITION - prišlo je do zadržane vrednosti. Podatki o zadržani vrednosti so na voljo v pomnilniku v strukturi ScsiPt, v kolikor je ta bil rezerviran;

SCSIPT_RETURN_ERROR - ukaz se ni izvedel, podatki o napaki so shranjeni v sistemski spremenljivki errno.

4. Posebnosti pri različnih sistemih

4.1. Microsoft windows

Windows od verzije NT uporabljajo dvonivojski gonilniški sistem za kontrolo naprav SCSI [4].. Na najnižjem nivoju je tako imenovani gonilnik priključka (*port driver*), na višjem nivoju pa je tako imenovani razredni gonilnik. Oba gonilnika sta del operacijskega sistema windows in se ob zagonu naložita avtomatsko.

Gonilnik priključka se odziva na množico zahtev, ki so bile poslane s strani sistema. Te zahteve so lahko poslane tudi kot zahteve, ki jih pošlje razredni gonilnik. Gonilnik priključka loči razredne gonilnike od specifičnih funkcij strojne opreme in časovno usklajuje operacije več razrednih gonilnikov. Gonilnik priključka pri SCSI podpira dostope več razrednih gonilnikov naenkrat, podpira pa tudi gonilnike, ki niso napisani posebej za naprave SCSI, ter s tem omogoča transparentnost fizičnih lastnosti dostopa do naprav.

Razredni gonilnik je višji nivo gonilnika, ki sicer prav tako podpira fizične lastnosti naprav, vendar pa morajo te vsebovati t. i. inteligentno kontrolo, kot na primer SCSI HBA (*host bus adapter*). Ta gonilnik komunicira s gonilnikom priključka in uporablja zahteve, ki so bile definirane na nivoju sistema. Windows uporabljajo razredni gonilnik SCSI za dostop do naprav SCSI.

Razredni gonilnik prav tako definira ime dostopa do naprav SCSI. Gonilnik ne definira posamično ime za vsako izmed priključenih naprav posebej, pač pa določi ime vmesniku SCSI. Tako dostopimo do vseh naprav, ki so priklopljene na isti vmesnik SCSI, preko istega opisnega parametra za datoteko. Poimenovanje pod Windows je določeno kot "scsiB:", kjer je B številka gonilnika SCSI-priključka oz. številka vodila na vmesniku.

3.1.1 Pošiljanje ukazov napravi

Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije CreateFile(). Nato se napolni struktura SCSI_PASS_THROUGH_DIRECT s podatki o ukazu SCSI. Posebnost pri tem klicu so parametri PathId, TargetId in Lun, ki se morajo definirati za vsak ukaz določeni napravi. V kolikor se ta polja ne bi definirala, razredni gonilnik ne bi vedel, do katere naprave dostopimo, ker ima samo podatke o vmesniku SCSI (*bus number*).

```
typedef struct _SCSI_PASS_THROUGH_DIRECT {
    USHORT Length;
    UCHAR ScsiStatus;
    UCHAR PathId;
    UCHAR TargetId;
    UCHAR Lun;
    UCHAR CdbLength;
    UCHAR SenseInfoLength;
    UCHAR DataIn;
    ULONG DataTransferLength;
    ULONG TimeOutValue;
    PVOID DataBuffer;
    ULONG SenseInfoOffset;
    UCHAR Cdb[16];
}SCSI_PASS_THROUGH_DIRECT, *PSCSI_PASS_THROUGH_DIRECT;
```

Struktura se pošlje na napravo z ukazom DeviceIoControl() kot poseben klic IOCTL_SCSI_PASS_THROUGH_DIRECT. V kolikor se je klic izvedel pravilno, vrne funkcija DeviceIoControl() pozitivno vrednost. Vendar pa to ne pomeni, da se je tudi ukaz izvedel pravilno. V kolikor je prišlo do zadržane vrednosti, bo klic prav tako uspel, status o napaki pa bo shranjen v parametru ScsiStatus strukture SCSI_PASS_THROUGH_DIRECT.

Če klic funkcije DeviceIoControl() ne uspe, potem to pomeni, da je prišlo do napake na sistemskem nivoju ali pa je gonilnik prestregel napako in jo interpretiral. Potem funkcija vrne vrednost 0 in podatki o napaki se dobijo s klicem funkcije GetLastError(). Primer takšne napake se zgodi, kadar je bila gonilniku že dana kakšna zahteva za ukaz, pri čemer vrne gonilnik napako in GetLastError() vrne vrednost DEVICE_BUSY.

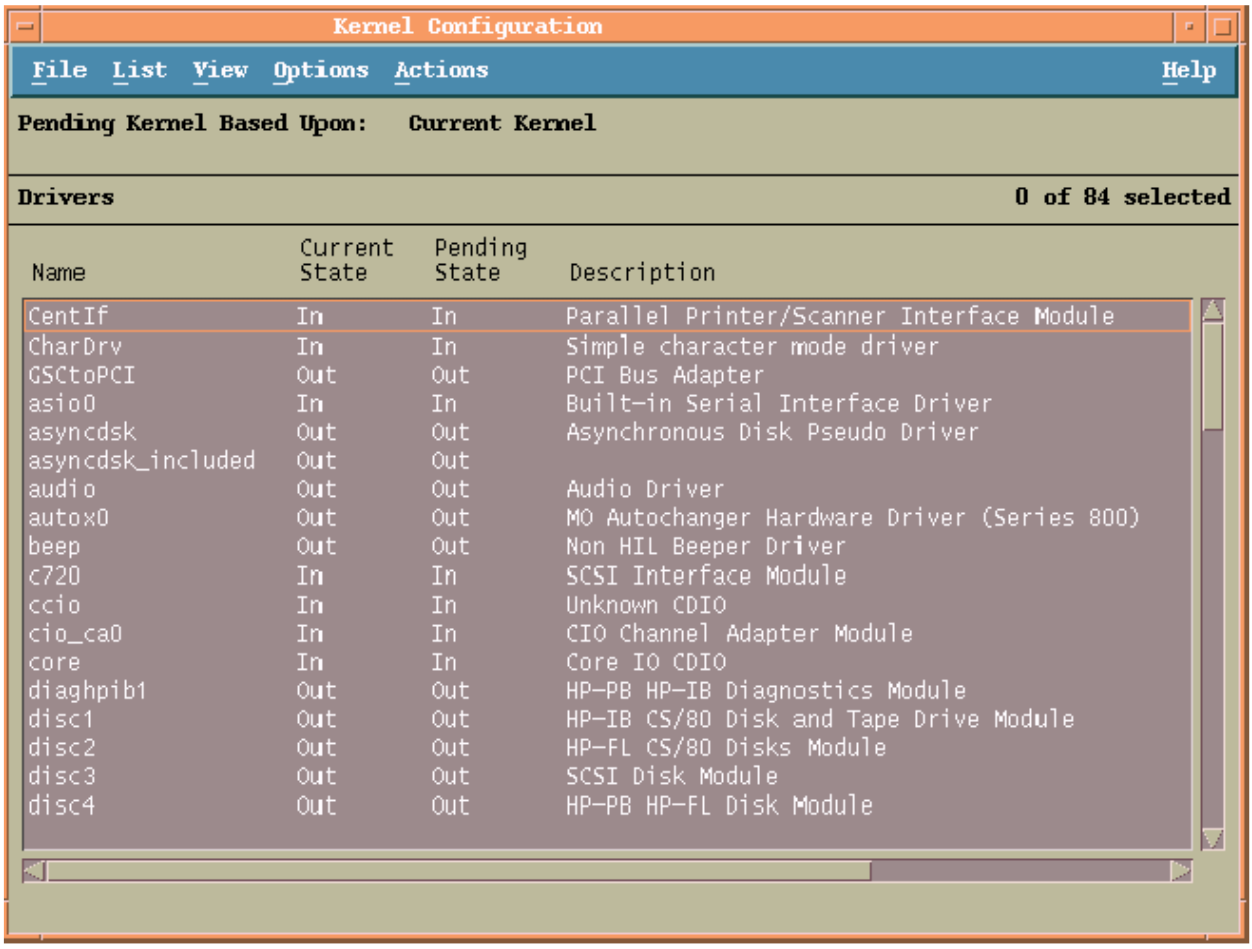
Do napake pride tudi, če se uporablja gonilnik za specifično napravo, ki si rezervira napravo v razrednem gonilniku za ekskluzivno uporabo. V tem primeru klic funkcije DeviceIoControl() vrne napako in GetLastError() vrne vrednost DEVICE_CLAIMED. Če je naprava rezervirana za gonilnik višjega nivoja, potem razredni gonilnik ne bo pustil želenega ukaza do naprave. Ponujata se dve možnosti. Prva je ta, da se gonilnik višjega nivoja izklopi, kar prepreči nadaljnjo uporabo naprave sistemu, druga pa je ta, da se uporabi gonilnik višjega nivoja za pošiljanje ukazov SCSI. Primer takšnega gonilnika je gonilnik za tračne naprave. Ta gonilnik ponuja možnost pošiljanja strukture SCSI_PASS_THROUGH_DIRECT s klicem DeviceIoControl() ter parametrom IOCTL_SCSI_PASS_THROUGH_DIRECT. Razlika je potemtakem samo v klicu funkcije

CreateFile(), kjer se namesto imena razrednega gonilnika "scsiB:" uporabi ime gonilnika za tračne enote. To ime ima format "TapeN", kjer je N primerek enote.

Gonilnik za tračne enote poimenuje svoje primerke po vrstnem redu (tape0, tape1 itd.). Številka primerka se poišče pod Start -> Control Panel -> Tape Devices -> Properties

4.2. UNIX HP-UX

Gonilniki pri sistemu HP-UX so prevedeni v jedru operacijskega sistema, kjer mora biti dodan tudi gonilnik za prepustnost ukazov SCSI. Ali je ta gonilnik dodan v jedru, lahko preverimo s pripomočkom SAM (*System Administration Manager*) po naslednjem postopku. Prijavimo se kot sistemski administrator v korenski imenik. V oknu 'System Administration Manager' izberemo 'Kernel Configuration' in potem 'Drivers'.



Slika 9: Okno SAM

Med naštetimi gonilniki mora biti napisan gonilnik z imenom 'sctl' ali 'spt'. Status tega gonilnika je označen v stolpcu 'Current State' in mora biti nastavljen na 'in'. V kolikor je gonilnik v stanju 'out', potem ga je treba dodati k jedru. V meniju 'Actions' izberemo opcijo 'Add To Kernel' in stanje v stolpcu 'Pending State' se mora spremeniti v 'in'. Nato v meniju 'Actions' izberemo 'Create a New

Kernel' in s tem se prevede jedro operacijskega sistema s spremembami, ki smo jih vnesli. Računalnik moramo ponovno zagnati, da se novo jedro prične izvajati [5].

Kot na vseh sistemih UNIX, tudi pod HP-UX dostopamo do naprave preko opisnega parametra za datoteke. Datoteko za naprave najdemo v imeniku /dev. Ponavadi se datoteke za naprave, ki so priključene na računalnik, namestijo same ob zagonu, v kolikor pa datoteka za našo napravo ne obstaja, jo moramo oblikovati sami. Preden pričnemo z oblikovanjem datoteke, moramo preveriti, ali je naprava pravilno priklopljena na sistem. To naredimo s ukazom '/usr/sbin/ioscan -f' (slika 10).

```
# ioscan -f
Class       I  H/W Path  Driver    S/W State H/W Type  Description
===========================================================================
bc          0             root      CLAIMED   BUS_NEXUS
ext_bus     0  52         scsi1     CLAIMED   INTERFACE HP 28655A - SCSI Interface
target      4  52.1       target    CLAIMED   DEVICE
disk        4  52.1.0     disc3     CLAIMED   DEVICE    SEAGATE ST15150N
target      1  52.2       target    CLAIMED   DEVICE
disk        0  52.2.0     disc3     CLAIMED   DEVICE    TOSHIBA CD-ROM XM-4101TA
target      3  52.4       target    CLAIMED   DEVICE
tape        0  52.4.0     tape2     CLAIMED   DEVICE    HP       C1533A
spt         1  52.4.1     spt       CLAIMED   DEVICE    HP       C1553A
target      6  52.5       target    CLAIMED   DEVICE
disk        5  52.5.0     disc3     CLAIMED   DEVICE    SEAGATE ST15150N
target      2  52.6       target    CLAIMED   DEVICE
disk        1  52.6.0     disc3     CLAIMED   DEVICE    SEAGATE ST15150N
lanmux      0  56         lanmux0   CLAIMED   INTERFACE LAN/Console
tty         0  56.0       mux4      CLAIMED   INTERFACE
lan         0  56.1       lan3      CLAIMED   INTERFACE
lantty      0  56.2       lantty0   CLAIMED   INTERFACE
processor   0  62         processor CLAIMED   PROCESSOR Processor
memory      0  63         memory    CLAIMED   MEMORY    Memory
# 
```

Slika 10: Izpis ioscan -f

V našem primeru imamo na sistem priključeno tračno enoto HP SureStore 12000e z robotiko za izmenjavanje medijev. Za tračno enoto se uporabi gonilnik 'stape', za katerega sistem ponavadi sam konfigurira datoteko. Za robotiko pa moramo generirati datoteko sami. Na sliki vidimo, da je vodilo SCSI kontrolirano s sistemskim gonilnikom 'scsi1'. Ta tip se imenuje vmesnik SCSI NIO in za naprave uporablja gonilnik 'sctl'. V kolikor bi sistem uporabljal drug vmesnik, bi se za naprave SCSI uporabil gonilnik 'spt' [6].

Za ustvarjanje datoteke potrebujemo nekatere informacije, in sicer glavno številko (*major number*) gonilnika in pa podštevilko (*minor number*), ki pa ni odvisna od tipa gonilnika. Glavno številko dobimo z ukazom 'lsdev -d sctl', podštevilko pa moramo sestaviti sami. Ima format tipa 0xIITL00, kjer sta II številka primerka vmesnika SCSI (ne naprave), T je naslov priključene naprave SCSI (v našem primeru robotike za menjavanje medijev), L je logično število enote (LUN), na koncu pa je treba dodati še dve ničli.

Tako lahko ustvarimo datoteko z ukazom "mknod /dev/spt/<ime> c Major# Minor#".

4.2.1. Pošiljanje ukazov SCSI

Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Kot parameter podamo tej funkciji datoteko, oblikovano, kakor je opisano v prejšnjem poglavju. Napolnimo strukturo sctl_io s podatki o ukazu SCSI in pokličemo funkcijo ioctl() s klicnim parametrom SIOC_IO.

```
struct sctl_io
{
      unsigned flags;               /* IN: SCTL_READ */
      unsigned cdb_length;          /* IN: dolžina ukaza */
      unsigned char cdb[16];        /* IN: opis ukaza */
      void *data;                   /* IN: pomnilnik za prenos podatkov */
      unsigned data_length;         /* IN: dolžina prenosa podatkov */
      unsigned max_msecs;           /* IN: milisekunde pred izločitvijo */
      unsigned data_xfer;           /* OUT: smer prenosa podatkov */
      unsigned cdb_status;          /* OUT: status SCSI */
      unsigned char sense[256];     /* OUT: pomnilnik zadržanih podatkov */
      unsigned sense_status;        /* OUT: status zadržanih podatkov SCSI */
      unsigned sense_xfer;          /* OUT: dolžina podatkov zadržane vr. */
      unsigned reserved[16];        /* IN: mora biti 0; OUT: nedoločeno */
};
```

V kolikor funkcija ioctl() vrne pozitivno vrednost, potem je bil ukaz uspešno predan napravi. Vrednost -1 pove, da je prišlo do napake, ki jo je gonilnik zaznal in ukaza ni izvedel. V kolikor je bil ukaz uspešno predan napravi, se po vrnitvi iz funkcije preveri vrednost spremenljivke cdb_status, ki pove, ali je prišlo do stanja zadržane vrednosti ali kakšne druge nenavadnosti. Posebnost za gonilnike pri sistemih so štiri dodatne vrednosti, ki pa jih v spremenljivko vrine gonilnik sam. Najpomembnejša izmed teh vrednosti je SCTL_INCOMPLETE, ki pove, da se ukaz

ni izvedel do konca in ga je treba poklicati znova. Druga takšna vrednost je SCTL_SELECT_TIMEOUT, ki pove, da je potekel čas, ki smo ga določili v spremenljivki sctl_io->max_msecs. To pomeni, da se v tem določenem času ukaz ni uspel izvesti in je gonilnik sam prekinil izvajanje ukaza. Tretja vrednost je SCTL_INVALID_REQUEST, ki pa je ne smemo zamenjati z zadržano vrednostjo 24:00-INVALID_REQUEST, kajti prva pomeni, da je napako zaznal gonilnik, torej mora biti nekaj narobe s podatki v strukturi sctl_io, ki so namenjeni gonilniku, medtem ko druga prihaja iz naprave same ter pomeni, da je napaka v polju CDB oz. ukazu samem. Zadnja posebna vrednost, ki jo gonilnik vrne, je SCTL_POWERFAIL, ki pove, da gonilnik ne more dostopiti do naprave.

Gonilnik za prenos ukazov SCSI ima lastnost, da se skupaj s statusom zadržane vrednosti vrnejo tudi podatki o njej. Tako jih ni treba dodatno zahtevati z ukazom REQUEST SENSE. Zato ima v strukturi rezerviran prostor za te podatke v polju sctl_io->sense. V spremenljivki sctl_io->sense_status je spravljeno stanje teh podatkov in v sctl_io->sense_xfer dolžina podatkov zadržane vrednosti.

4.3. UNIX Sun-Solaris

Operacijski sistem Solaris v osnovi ne vsebuje gonilnika za izvajanje ukazov SCSI, zato je treba v te namene na sistem nastaviti poseben gonilnik, ki se imenuje 'sst'. Ker gonilniki niso povezani v samo jedro operacijskega sistema, je postopek dodajanja gonilnika precej preprost: poskrbeti je treba le za namestitvene korake, ki pa se ne izvedejo avtomatično, pač pa jih moramo postoriti sami. Najprej je treba namestiti datoteko z imenom 'sst' in konfiguracijsko datoteko 'sst.conf' v imenik '/kernel/drv/'. V sistemsko datoteko '/etc/devlink.tab' je treba dodati vrstico

“ type=sample_driver;name=sst;minor=character rsst\A1”

Ta dodatek bo naročil sistemu, naj avtomatsko generira posebno datoteko za napravo SCSI z imenom '/dev/rsstX', kjer X predstavlja SCSI-identifikacijo priključene naprave SCSI. Posebej je treba paziti, da se v to datoteko ne vstavi znak za presledek, pač pa je v te namene treba uporabiti tabulator.

Popraviti je treba še datoteko 'sst.conf', in sicer dodati oz. odkomentirati vrstico, ki določa identifikacijo priključene naprave SCSI. Za SCSI ID 4 na primer bi odkomentirali vrstico

name=”sst” class= “scsi”
target=4 lun=0;

Da sistem prepozna priključeno napravo, je treba ponovno zagnati računalnik ter preveriti, ali je naprava zaznana pravilno. To se naredi najprej z ukazom 'shutdown -i0', nato pa ob zagonu v pozivni vrstici 'ok' napišemo 'probe-scsi', nato pa 'boot -r'. Ko se sistem znova zažene, imamo v našem primeru ustvarjeno datoteko '/dev/rsst4', ki jo lahko uporabimo kot datoteko za pošiljanje ukazov SCSI [9].

4.3.1. Pošiljanje ukazov SCSI

Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Napolnimo strukturo uscsi_cmd in pokličemo sistemsko funkcijo ioctl() s parametrom USCSICMD.

```
struct uscsi_cmd {
```

```
        int         uscsi_flags;      /* branje, pisanje, ... */
        short       uscsi_status;     /* končni status  */
        short       uscsi_timeout;    /* čas čakanja, da se ukaz izvede */
        caddr_t     uscsi_cdb;        /* opis ukaza */
        caddr_t     uscsi_bufaddr;    /* pomnilnik s podatki */
        u_int       uscsi_buflen;     /* velikost pomnilnika s podatki */
        u_int       uscsi_resid;      /* resid vhodno-izhodne operacije */
        u_char      uscsi_cdblen;     /* velikost ukaza */
        u_char      uscsi_rqlen;      /* velikost uscsi_rqbuf */
        u_char      uscsi_rqstatus;   /* status ukaza 'request sense' */
        u_char      uscsi_rqresid;    /* resid ukaza 'request sense' */
        caddr_t     c;                /* pomnilnik podatkov zadržane vrednosti */
        void        *uscsi_reserved_5; /* rezervirano za razširitve */
};
```

Posebnost pri tej strukturi glede na ostale sisteme je v tem, da v njej ni rezerviranega pomnilnika za polje ukaza (*CDB field - command descriptor block*), pač pa ga moramo rezervirati sami in nastaviti nanj kazalec uscsi_cmd->uscsi_cdb.

V kolikor funkcija ioctl() vrne pozitivno vrednost, potem se je ukaz uspešno izvedel. V kolikor funkcija vrne vrednost -1, to pomeni, da je prišlo do napake in da ukaz ni bil predan napravi. Razlog za napako je shranjen v sistemski spremenljivki errno. Če se je ukaz izvedel, vendar je prišlo do zadržane vrednosti, so podatki o njej že shranjeni v rezerviranem pomnilniku uscsi_cmd->uscsi_rqbuf. Ta pomnilnik moramo rezervirati sami, drugače moramo podatke o zadržani vrednosti zahtevati naknadno z ukazom REQUEST SENSE.

4.4. UNIX IBM AIX

Operacijski sistem AIX [9] v osnovi nima priloženega gonilnika za izvajanje ukazov SCSI, zato je treba namestiti dodaten gonilnik, ki ga ima v svoji ponudbi programski paket netvault. Netvault je v osnovi programski paket za arhiviranje podatkov, vendar pa mu je dodan tudi gonilnik za izvajanje ukazov SCSI, ki ga lahko uporabimo v lastne namene.

4.4.1. Priklop naprav

Za priklop naprave na sistem AIX je treba sistem najprej ugasiti, napravo priklopiti in ponovno zagnati sistem. Preden se sistem zažene, je treba preveriti, ali priklopljena naprava ne uporablja enak ID, kot že kakšna prej priključena naprava. Po zagonu in instalaciji gonilnika je mogoče lastnosti ter nastavitve priklopljenih naprav spreminjati s priloženim orodjem SMIT (*System Management Interface Tool*) [9].

4.4.2. Pošiljanje ukazov SCSI

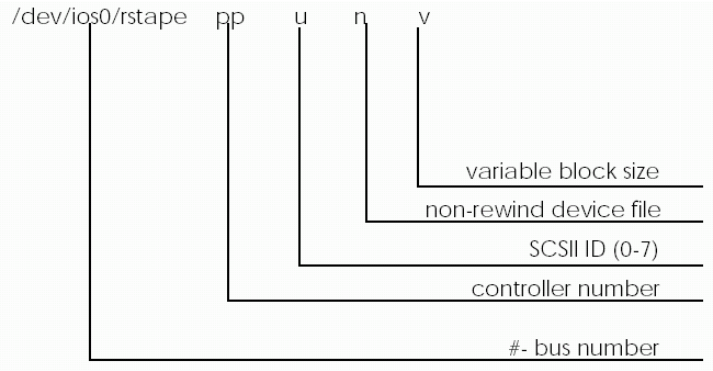
Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Napolnimo strukturo sc_iocmd ter pokličemo sistemsko funkcijo ioctl() s parametrom STIOCMD.

```
struct sc_iocmd {
      uint   data_length;          /* dolžina podatkov za prenos */
      char   *buffer;              /* kazalec na pomnilnik s podatki */
      uint   timeout_value;        /* trajanje ukaza, podano v sekundah */
      uchar  status_validity;      /* veljavnost statusa */
      uchar  scsi_bus_status;      /* status vodila (če veljaven) */
      uchar  adapter_status;       /* status priključka SCSI (če veljaven) */
      uchar  resvd1;               /* rezervirano, naj bo 0 */
      uchar  resvd2;               /* rezervirano, naj bo 0 */
      uchar  adap_q_status;        /* vračanje statusa priključka */
      uchar  q_tag_msg;            /* položaj v vrsti */
      uchar  flags;                /* SC_NODISC, SC_ASYNC, B_READ, B_WRITE */
      uchar  q_flags;              /* pove mesto v vrsti priključki SCSI */
      uchar  lun;                  /* logično število enote     */
      uchar  resvd7;               /* rezervirano, naj bo 0 */
      uchar  command_length;       /* dolžina ukaza */
      uchar  scsi_cdb[12];         /* polje za opis ukaza */
};
```

Pri sistemih IBM AIX deluje klic funkcije ioctl() drugače kot pri ostalih sistemih UNIX. Funkcija ob uspešno izvedenem ukazu vrne vrednost 0, drugače pa negativno vrednost in oznaka napake je shranjena v sistemski spremenljivki errno. Posebnost je tudi ta, da v kolikor pride do zadržane vrednosti na napravi, funkcija prav tako vrne negativno vrednost ter sistem označi napako. Zato je ob neuspelem klicu treba dodatno preveriti sistemsko spremenljivko errno za vrednost EIO, ki pomeni, da je prišlo do napake pri vhodno-izhodni operaciji. Nato moramo preveriti vrednost bitov v spremenljivki sc_iocmd->status_validity. V kolikor je nastavljen bit SC_ADAPTER_ERROR, to pomeni, da je prišlo do dejanske napake pri komunikaciji in iniciator ni mogel vzpostaviti povezave z napravo. V kolikor pa je nastavljen bit SC_SCSI_ERROR, to pomeni, da je prišlo do zadržane vrednosti. Podatke o zadržani vrednost dobimo z ukazom REQUEST SENSE.

4.5. UNIX Siemens SINIX

Gonilnik za izvajanje ukazov SCSI je vgrajen v operacijski sistem. SINIX ima poseben način imenovanja posebnih datotek za naprave SCSI. Datoteke se ustvarijo ter izpišejo z ukazom 'autoconf -l'. Format zapisa datoteke ja naslednji (primer za gonilnik tračnih naprav):



Slika 111: Format zbirk pri sinixu

Poiskati je treba datoteko, ki se sklada s podatki priključene naprave SCSI, in jo uporabiti za lastne namene.

4.5.1. Pošiljanje ukazov SCSI

Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Napolnimo strukturo sgen_escsicmd in pokličemo sistemsko funkcijo ioctl() s parametrom SGEN_ESCSICMD.

```
struct      sgen_escsicmd {
      u_char      scsi0;            /* zlog 0 */
      u_char      scsi1;            /* zlog 1 */
      u_char      scsi2;            /* zlog 2 */
      u_char      scsi3;            /* zlog 3 */
      u_char      scsi4;            /* zlog 4 */
      u_char      scsi5;            /* zlog 5 */
      u_char      scsi6;            /* zlog 6 */
      u_char      scsi7;            /* zlog 7 */
      u_char      scsi8;            /* zlog 8 */
      u_char      scsi9;            /* zlog 9 */
      u_char      scsi10;           /* zlog 10 */
      u_char      scsi11;           /* zlog 11 */
      int         scs_flags;        /* status */
      char        *scs_addr;        /* kazalec na pomnilnik s podatki */
      int         scs_bytes;        /* količina prenosa podatkov */
```

```
      int         scs_timeout;      /* čas v sekundah */
      int         scs_cdblen;       /* dolžina ukaza */
      short       scs_errtype;      /* tip napake */
      u_char      scs_scstat;       /* status SCSI */
      u_char      scs_shaier;       /* status priključka */
      int         scs_snslen;       /* dolžina podatkov zadržane vrednosti */
      char        *scs_snsaddr;     /* kazalec na podatke zadržane vrednosti */
      int         dummy[5];         /* rezervirano za razširitve */
};
```

Gonilnik za prenos ukazov SCSI na naprave ponuja možnost, da se skupaj s statusom zadržane vrednosti vrnejo tudi podatki o njej. Tako jih ni treba dodatno zahtevati z ukazom REQUEST SENSE. Zato ima v strukturi rezerviran kazalec za te podatke v polju sgen_escsicmd->scs_snsaddr. V spremenljivki sgen_escsicmd->scs_snslen je shranjena dolžina podatkov zadržane vrednosti. Ta možnost je opcijska in se vklopi z dodatnim bitom SGENAUTORQS v spremenljivki sgen_escsicmd->scs_flags.

4.6. Linux

Operacijski sistem Linux ima v osnovi priložen gonilnik za izvajanje ukazov SCSI, in dodatni gonilnik, imenovan SG. Gonilnik SG deluje na poseben način, in sicer ne uporablja standardnega kontrolnega ukaza za vpeljevanje ukaza SCSI ioctl(), temveč uporablja ukaza write() in read().

4.6.1. Priklop naprav

V datoteki `/proc/scsi/scsi` lahko vidimo, kateri SCSI ID je še prost. Sistem ugasimo, priklopimo napravo ter ponovno zaženemo. Znova lahko preverimo datoteko `/proc/scsi/scsi`, da se prepričamo, ali je bila naprava s strani sistema prepoznana. Pri zagonu računalnika bo operacijski sistem sam oblikoval datoteke za priključene naprave. Z ukazom "`dmesg | grep scsi`" dobimo izpis datotek za priključene naprave SCSI [8].

4.6.2. Pošiljanje ukazov SCSI skozi gonilnik SG

Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Rezervira se pomnilnik, ki vsebuje glavo ukaza, ukaz in podatke, ki se prenašajo, če se. Najprej se napolni struktura sg_header:

```
struct sg_header
 {
   int pack_len;                    /* dolžina paketa (vključno z glavo) */
   int reply_len;                   /* največja dolžina pričakovanega odziva */
   int pack_id;                     /* identifikacija paketa */
   int result;                      /* 0==ok, drugače št. napake */
   unsigned int twelve_byte:1;      /* ali ima ukaz 12 zlogov? */
   unsigned int other_flags:31;     /* rezervirano */
   unsigned char sense_buffer[16];  /* se uporabi samo pri branju podatkov */
};
```

Strukturi sledijo ukaz in podatki za ukaz, nato pokličemo ukaz write(). V primeru, ko smo podatke pisali na napravo, pri uspešnem zapisu vrnjena vrednost klica ne vsebuje napake, v kolikor pa smo podatke iz naprave brali, klic funkcije vrne napako EINTR, ko so podatki pripravljeni za branje. Klic funkcije read() nato vrne podatke, ki so bili prebrani iz naprave.

4.6.3. Pošiljanje ukazov SCSI skozi gonilnik za izvajanje ukazov

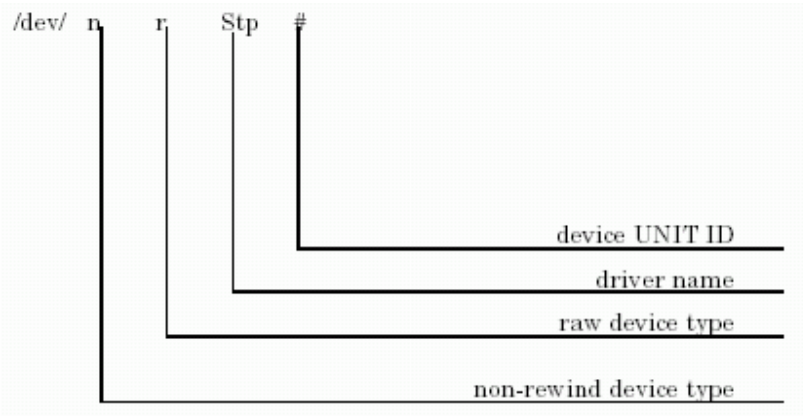
Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Rezervira se pomnilnik, v katerega se vpišejo opis ukaza in podatki, ki se prenašajo, če se. Nato se pokliče sistemska funkcija ioctl() s parametrom 1 (ime ni definirano). Vrnjena vrednost klica vsebuje status prenosa (ne samo uspešnost klica, kot je to v navadi pri drugih sistemih UNIX). V kolikor je status manjši od nič, potem je bil klic neuspešen in sistemske spremenljivka errno vsebuje opis napake. V kolikor pa je status večji ali enak nič, lahko preverimo njegovo vrednost in se nanjo odzovemo. Spremenljivka vsebuje del, ki ga napolni gonilnik, in del, ki govori o statusu ukaza SCSI. Gonilniški del vsebuje opis uspešnosti prenosa ter priporočilo, kako nadaljevati. Priporočilo je lahko takšno, ki priporoča prekinitev ali pa priporoča ponoven poskus. Del, ki vsebuje status ukaza SCSI, pove, kaj je v vrnjenem pomnilniku. V kolikor je bil ukaz uspešno izveden in je prebral podatke, potem so v vrnjenem pomnilniku podatki, ki jih je gonilnik prebral iz naprave, v kolikor pa je prišlo do zadržane vrednosti, vrnjen pomnilnik vsebuje podatke o njej. Pri ponovnem poskusu vzpostavitve ukaza moramo biti previdni, da se pomnilnik znova inicializira, saj je bil morda prepisan z vsebino zadržane vrednosti [7].

4.7. UNIX SCO

Operacijski sistem SCO vsebuje v svojem jedru gonilnik za izvajanje ukazov SCSI. Pri vzpostavitvi sistema se v imeniku /dev oblikujejo opisne datoteke za priključene naprave SCSI. SCO ima neke vrste šolski primer gonilnika za ukaze SCSI, saj ne vsebuje nobenih posebnosti in ima strukturo za prepustnost ukazov dobro organizirano.

4.7.1. Priklop naprav

V datoteki /etc/conf/cf.d/mscsi lahko vidimo, kateri SCSI ID je še prost. Sistem ugasimo, priklopimo napravo ter ponovno zaženemo. Znova lahko preverimo datoteko /etc/conf/cf.d/mscsi, da se prepričamo, ali je bila naprava s strani sistema prepoznana. Nato moramo napisati ukaz mkdev <gonilnik>, kjer je gonilnik "tape" za tračne enote ali "juke" za robotiko. Prav tako obstajajo gonilniki za ostale naprave SCSI, ki podpirajo prepustnost ukazov SCSI. Oblikovala se bo datoteka, ki ima naslednji format



Slika 122: Format zbirke pod sistemom SCO

4.7.2. Pošiljanje ukazov SCSI

Napravi SCSI se pošiljajo ukazi preko opisnega parametra za datoteko, ki se mora kreirati s klicem funkcije open(). Napolnimo strukturo scsicmd2 in pokličemo sistemsko funkcijo ioctl() s parametrom SCSIUSERCMD2.

```
struct scsicmd {
        faddr_t data_ptr;       /* pomnilnik za podatke */
        ulong   data_len;       /* dolžina prenosa podatkov   */
        ushort  is_write;       /* smer prenosa podatkov: 0=branje, 1=pisanje */
        uchar_tcdb[SCSICMDLEN];/* opis ukaza    */
        uchar_thost_sts;        /* status priključka SCSI       */
        uchar_ttarget_sts;      /* status naprave SCSI          */
        uchar_tcdb_len;         /* za lastno uporabo proizvajalcem  */
};

struct scsicmd2 {
        struct scsicmd cmd;     /* Ukaz SCSI */
        int     sense_len;      /* dolžina podatkov zadržane vrednosti */
        void    *sense_ptr;     /* pomnilnik zadržanih podatkov  */
};
```

V kolikor funkcija ioctl() vrne pozitivno vrednost, potem se je ukaz uspešno izvedel. V kolikor funkcija vrne vrednost -1, to pomeni, da je prišlo do napake in da ukaz ni bil predan napravi. Razlog za napako je shranjen v sistemski spremenljivki errno. Če se je ukaz izvedel, vendar je prišlo do zadržane vrednosti, so podatki o njej že shranjeni v rezerviranem pomnilniku uscsi_cmd->uscsi_rqbuf. Ta pomnilnik moramo rezervirati sami, drugače moramo podatke o zadržani vrednosti zahtevati naknadno z ukazom REQUEST SENSE.

4.8. UNIX Dynix

Operacijski sistem Dynix ne vsebuje gonilnika za izvajanje ukazov SCSI. V kolikor bi hoteli pošiljati ukaze v napravo SCSI, bi si morali gonilnik napisati sami, zato bomo tukaj opisali način, kako gonilniki višjega nivoja dostopijo do naprav SCSI. Sistem ima vgrajen vmesnik do naprav SCSI, ki ga uporabljajo gonilniki višjega nivoja, kot je npr. gonilnik za tračne enote.

3.8.1 Pošiljanje ukazov SCSI

Gonilnik uporabi klic sdioctl() za vzpostavitev vhodno-izhodne operacije. Parameter za ta klic je rezerviran pomnilnik, katerega glava je struktura scsiioctl, sledijo pa podatki za prenos.

```
struct scsiioctl {
        ulong sio_datalength;
        ulong sio_addr;
        ulong sio_cmdlen;
        union {
                struct cmd6 cmd6;
                struct cmd10 cmd10;
                unchar cmd[12];                 /* Max SCSI command size */
        } sio_cmd;
};
```

Vrnjena vrednost pove uspešnost klica. Žal ni znano nič izrecnega o obnašanju te funkcije pri sistemu dynix.

5. Sklep

Opisali smo standard SCSI-2 in podali kratek pregled, kaj vse zajema ta standard. S tem smo pripravili izhodišče za razlago funkcionalnosti, ki smo jo morali vgraditi v vmesnik za zagotavljanje sistemske neodvisnosti. Nadalje smo definirali splošni pristop za izvedbo ukazov SCSI in pokazali, kako se izvedejo pri različnih računalniških sistemih. Napisali smo programski vmesnik za aplikacijo HP Data Protector, kjer je trenutno uporabljen v okolju arhiviranja podatkov brez podatkovnega strežnika. Izkazal se je kot uporaben in ga nameravamo integrirati tudi v aplikacijo za krmiljenje robotske roke pri izmenjevalnikih medijev, in sicer za dostop do tračnih enot in za dostop do magnetno-optičnih diskov.

Izvedbo naloge je omogočilo podjetje Hermes Softlab-u, ki je zagotovilo dostop do strojne opreme in do interneta, kjer smo zbrali pretežno večino dokumentacije za svoje razvojno delo.

Viri, Literatura:

[1] Small Computer System Interface-2, X3T9.2/375R revision 10L

[2] SCSI-3 Architecture Model, ANSI X3.270-1996

[3] SCSI Architecture and drafts, T10 Comitee, http://www.t10.org/

[4] Microsoft Developer Network, http://msdn.microsoft.com

[5] Stran pomoči HP-UX: scsi(7) - Small Computer System Interface device drivers

[6] Stran pomoči HP-UX: scsi_ctl(7) - SCSI pass-through driver for sdisk, schgr, sflop, stape device drivers

[7] The Linux 2.4 SCSI subsystem, Douglas Gilbert, November 2002, http://www.linuxdocs.org/HOWTOs/SCSI-2.4-HOWTO/index.html

[8] The Linux Documentation Project, December 2002, http://www.tldp.org/

[9] AIX Version 4.3 Extended Documentation, Januar 2003, http://nscp.upenn.edu/aix4.3html/aixgen/wxinfnav/topnav.htm