developer-PL

Grzegorz Krashan <krashan@matay.pl>

COLLABORATORS

	TITLE : developer-PL	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Grzegorz Krashan <krashan@matay.pl>	January 23, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

developer-PL

1.1 Spis treôci

P R O M E T E U S Z

Informacje dla programistów

Wersja 1.48 (24 maja 2001)

Spis treôci:

1.2 Wprowadzenie do PCI

Podstawowe informacje o PCI

W tym rozdziale znajdziecie podstawowe informacje niezbędne z punktu widzenia programisty dotyczâce samego PCI. PCI (Peripherial Component Interface) to uniwersalna 32-bitowa magistrala sîuûâca do îâczenia między sobâ procesora i urzâdzeï peryferyjnych umieszczanych wewnâtrz komputera. Duûe rozpowszechnienie PCI w komputerach PC zaowocowaîo ogromnâ iloôciâ dostępnych na rynku tanich kart rozszerzeï. Prometeusz jest jednym z rozwiâzaï pozwalajâcych na uûywanie tych kart w Amidze.

Na magistrali PCI istniejâ dwie 32-bitowe przestrzenie adresowe - przestrzeï wejôcia/wyjôcia i przestrzeï pamięci. Daje to îâcznie 8 GB przestrzeni adresowej. Nazwy tych przestrzeni sâ umowne, z reguîy karty umieszczajâ w przestrzeni I/O swoje rejestry kontrolne, a w przestrzeni pamięci bufory pamięciowe (np. pamięê video karty graficznej). Czësto rejestry kontrolne sâ zdublowane w obu obszarach. Obszar w przestrzeni I/O moûe mieê od 4 bajtów do 4 GB, obszar pamięci od 16 bajtów do 4 GB. Oczywiôcie w ûadnej Amidze nie mamy do dyspozycji 8 GB przestrzeni adresowej, dlatego Prometeusz udostępnia jedynie jej fragment, a dokîadniej 511 MB obszaru pamięci i 960 kB obszaru wejôcia/wyjôcia (patrz mapa pamięci).

Kaûda karta PCI przed jej uûyciem wymaga konfiguracji. Podstawowym zadaniem konfiguracji jest dynamiczne przydzielenie adresów. Karta PCI moûe zaûâdaê od systemu przydzielenia maksymalnie szeôciu obszarów adresowych w dowolnej z dwóch przestrzeni adresowych. W systemie AmigaOS przydzielenie adresów wykonuje automatycznie biblioteka prometheus.library w czasie startu systemu. Sterowniki poszczególnych kart mogâ je odszukaê i odczytaê ich adresy bazowe korzystajâc z funkcji tej biblioteki.

1.3 Mapa przestrzeni adresowej

Podziaî przestrzeni adresowej Prometeusza

Podane tu informacje nie sâ potrzebne dla programistów piszâcych sterowniki dla AmigaOS - ci powinni korzystaê z funkcji dostarczanych przez bibliotekë prometheus.library. Znajomoôê mapy pamiëci Prometeusza moûe byê jednak niezbëdna do napisania sterowników dla innych systemów operacyjnych, na przykîad Linuksa czy NetBSD.

Prometeusz zajmuje 512 MB przestrzeni adresowej Amigi. Jej adres zaleûy od tego, jakie inne karty Zorro III sâ zainstalowane w systemie. W przypadku braku innych kart Prometeusz jest umieszczany przez system w zakresie adresów $40000000 - $5FFFFFFF. Nigdy jednak nie naleûy zakîadaê, ûe jest to adres staîy, zawsze naleûy go odczytaê z dynamicznych tablic tworzonych przez dany system operacyjny.

Jak wiadomo z wczeôniejszych rozdziaîów, PCI ma trzy przestrzenie adresowe, które muszâ zostaê jakoô zmieszczone w jednej przestrzeni adresowej Amigi. Dlatego róûnym fragmentom przestrzeni adresowej Prometeusza odpowiadajâ róûne przestrzenie adresowe PCI. Poniûej przedstawiona jest mapa pamiëci. Podane adresy sâ to przesuniëcia w stosunku do adresu bazowego karty:

$00000000 - $000EFFFF Przestrzeï wejôcia/wyjôcia PCI (960 kB)

$000F0000 - $000F00FF Przestrzeï konfiguracyjna slotu 0 (256 B)

$000F0100 - $000F1FFF Zarezerwowane

$000F2000 - $000F20FF Przestrzeï konfiguracyjna slotu 1 (256 B)

$000F2100 - $000F3FFF Zarezerwowane

$000F4000 - $000F40FF Przestrzeï konfiguracyjna slotu 2 (256 B)

$000F4100 - $000F5FFF Zarezerwowane

$000F6000 - $000F60FF Przestrzeï konfiguracyjna slotu 3 (256 B)

$000F6100 - $000FFFFF Zarezerwowane

$00100000 - $1FFFFFFF Przestrzeï pamiëci PCI (511 MB)

Przydzielajâc obszary adresów kartom PCI naleûy pamiëtaê o prawidîowym przypisaniu przestrzeni adresowej w zaleûnoôci od wymagaï karty. Przypomnë raz jeszcze, ûe w systemie AmigaOS zajmuje sië tym biblioteka prometheus.library.

1.4 Problem kolejnoôci bajtów

Problem kolejnoôci bajtów

Programujâc karty PCI zetkniemy sië z problemem kolejnoôci bajtów w sîowach 16 i 32-bitowych. Wiëkszoôê istniejâcych procesorów, z rodzinâ M68k i PPC wîâcznie uûywa konwencji zgodnie z którâ bajty w sîowie umieszczone sâ od najbardziej do najmniej znaczâcego:

0123

bity 31-24bity 23-16bity 15-8bity 7-0

Niestety w procesorach kompatybilnych z seriâ x86 Intela bajty umieszczane sâ w pamiëci odwrotnie:

0123

bity 7-0bity 15-8bity 23-16bity 31-24

Poniewaô wiëkszoôê kart PCI produkowana jest z myôlâ o komputerach PC, spodziewajâ sië one wîaônie takiego uszeregowania bajtów. Aby uniknâê zamieszania Prometeusz wyposaûony jest w sprzëtowy ukîad zmiany kojelnoôci bajtów, który caîâ operacjë wykonuje w locie bez narzutu czasowego. Dziëki temu moûemy programowaê karty PCI dokîadnie w taki sam sposób w jaki robi sië to na komputerach PC. Ukîad zamiany dziaîa w obu kierunkach. Jeûeli zatem w 32-bitowym rejestrze karty chcemy zapisaê danâ $DEADBACA, to robimy to bezpoôrednio, w kolejnych bajtach rejestru znajdâ sië $CA, $BA, $AD, $DE (zgodnie z konwencjâ Intela), ale odczyt z tego rejestru ponownie da nam $DEADBACA.

Autorzy sterowników dla systemów innych niû AmigaOS natknâ sië na jeden wyjâtek. Mianowicie rejestry konfiguracyjne kart PCI zgodnie ze specyfikacjâ PCI 2.1 majâ bajty uszeregowane w konwencji Motoroli. Ponieważ ukîad zmiany kolejnoôci bajtów w Prometeuszu pracuje caîy czas, naleûy przy dostëpie do przestrzeni konfiguracyjnej dokonywaê programowej zmiany kolejnoôci bajtów, która zneutralizuje zamianë sprzëtowâ. Programowâ zmianë kolejnoôci moûna wykonaê np. takâ sekwencjâ rozkazów asemblera:

ROL.W #8,d0

SWAP d0

ROL.W #8,d0

Odpowiednie makra (swapl() i swapw()) dla kompilatora GCC znajdujâ sië w inkludach doîâczonych do tego SDK. W systemie AmigaOS konfiguracjâ kart zajmuje sië prometheus.library, wiëc problem ten nie istnieje.

1.5 Podstawy dziaîania Prometeusza

Podstawy dziaîania Prometeusza

Prometeusz jest mostkiem miëdzy magistralami Zorro III i PCI. Jest to tak zwany mostek "przezroczysty". Kaûda transakcja Zorro III jest tîumaczona na odpowiedniâ transakcjë PCI. Przezroczystoôê oznacza, ûe nie sâ potrzebne ûadne specjalne funkcje do odczytu i zapisu danych do kart PCI. Karty te moûna traktowaê jako obszary przestrzeni adresowej procesora Amigi. Naleûy jednak zwróciê uwagë na pamiëê podrëcznâ (cache) procesora. W systemie AmigaOS procedury AutoConfig automatycznie wyîâczajâ cache procesora dla obszaru zajmowanego przez Prometeusza, niemniej jest to moûliwe tylko dla procesorów wyposaûonych w ukîad MMU. Problemy mogâ wystâpiê z procesorem 68EC030, nie ma on MMU wiëc pamiëê podrëcznâ w razie potrzeby naleûy opróûniê uûywajâc funkcji CacheClearU() lub lepiej CacheClearE() czyszczâc jedynie pamiëê podrëcznâ danych.

1.6 Biblioteka prometheus.library i jej zadania

Biblioteka prometheus.library i jej zadania

Biblioteka prometheus.library jest dodawana do systemu przez procedury AutoConfig w czasie startu systemu. Dokîadnie odbywa sië to w momencie wywoîania komendy BindDrivers w startup-sequence. Biblioteka wykrywa umieszczone w slotach Prometeusza karty PCI i konfiguruje je. Nastëpnie pozostaje w systemie umoûliwiajâc odszukanie dostëpnych kart PCI i pobranie informacji o nich. Do sprawdzenia, czy dana karta jest zainstalowana sîuûy funkcja Prm_FindBoardTagList(). Informacje o odszukanej karcie (w tym adresy i rozmiary przydzielonych obszarów przestrzeni adresowej) moûna odczytaê funkcjâ Prm_GetBoardAttrsTagList(). Obie funkcje szczegóîowo omówione sâ w pliku autodoc biblioteki. Przykîadem programu korzystajâcego z funkcji prometheus.library jest PrmScan wypisujâcy informacje o wszystkich znalezionych kartach PCI. Jego kod úródîowy znajduje sië w tym SDK.

1.7 Problemy z wolnymi kartami

Problemy z wolnymi kartami PCI

Specyfikacja PCI zaleca, aby kaûda karta rozpoczëîa operacjë zapisu lub odczytu najpóúniej w 8 taktów zegara od inicjalizacji cyklu PCI. Niestety niektóre karty w okreôlonych sytuacjach nie sâ w stanie tego zrobiê. Dotyczy to na przykîad odczytu zawartoôci pamiëci ROM niektórych kart graficznych. W takiej sytuacji karta sygnalizuje stan "Retry" proszâc inicjatora transakcji o powtórzenie cyklu. Niestety nie zawsze jest to moûliwe w obrëbie jednego cyklu szyny Zorro III. Konstrukcja Amigi nakîada ograniczenie na dîugoôê cyklu Zorro III, wynoszâce okoîo 1 mikrosekundy. Jeûeli w tym czasie cykl sië nie zakoïczy, jest on przerywany i generowany jest bîâd magistrali. Aby nie dopuôciê do zawieszenia systemu Prometeusz nie powtarza cyklu na ûâdanie "Retry", zwracajâc (w wypadku operacji odczytu) wartoôê $FFFFFFFF. W takim przypadku naleûy operacjë odczytu powtórzyê. Rozwiâzanie takie pokazane jest w kodzie úródîowym programu RomDump.

UWAGA:

Jak na razie jedyny stwierdzony przypadek opisanej sytuacji to odczyt pamiëci ROM karty Voodoo3.