WinSTon V 0.53

PCI-Support für RTOS-UH

IEP GmbH • Am Pferdemarkt 9c • D-30853 Langenhagen • Tel.: +49 (511) 70832-0 • Fax: +49 (511) 70832-99 • E-Mail: info@iep.de
Web: http://www.iep.de

1 Überblick

WinSTon ist ein von Paul Bates geschriebener ATARI-ST-Emulator, unter dem RTOS-UH bei Verzicht auf die Echtzeiteigenschaften des Betriebssystems lauffähig ist. Damit sind sowohl ein Kennenlernen des Betriebssystems RTOS-UH als auch erste PEARL-Programmierschritte direkt auf einem handelsüblichen PC möglich.

Mit den Änderungen der Version 0.53 wird auch die Nutzung handelsüblicher PCI-IO-Karten, z.B. für digitale und analoge I/O, auf dem PC unterstützt. Damit können auch erste praktische Schritte beim Einsatz von PEARL direkt auf dem PC gegangen werden – immer noch unter Verzicht auf die für den tatsächlichen Einsatz entscheidende Echtzeitfähigkeit von RTOS-UH.

Dieses Dokument beschreibt die Grundlagen für die Nutzung der PCI-Unterstützung und soll als Hilfe bei der PEARL-Programmierung dienen. Direkt unterstützt werden z.Zt. PCI-Karten der Fa. Kolter Electronic, Steinstr. 22, 50374, Erftstadt, www.kolter.de. Als Beispiel dient eine A/D-D/A-Karte.

2 Portzugriffe unter Windows

Alle Betriebssysteme der Windows-NT-Familie (NT4, W2K, XP) verweigern normalen Anwenderprogrammen den direkten Zugriff auf den PCI-Bus. Anders, als in früheren Systemen, ist dieser Zugriff nur Treibern gestattet.

Handelsübliche PCI-Karten werden daher stets mit einem Treiber ausgeliefert. Der Treiber besteht üblicherweise aus mindestens zwei Komponenten: dem eigentlichen Treiber (einer .sys/.vxd-Datei) und einer Treiber-Dll, die mit dem Anwenderprogramm geladen wird. Das Anwenderprogramm ruft Funktionen aus der Treiber-Dll auf, diese rufen die Funktionen des eigentlichen Treibers auf. Der Treiber selbst muss installiert, d.h. dem System als Gerätetreiber bekannt gemacht werden.

2.1 Treiberzugriff unter WinSTon

In WinSTon V0.53 wurde eine Möglichkeit integriert, über einen Gemdos-Call (Systemaufruf) eine Treiber-Dll zu laden und Funktionen aus dieser Dll aufzurufen. Die Treiber-Dll wird erst bei Bedarf dynamisch geladen, d.h. solange die Dll nicht benötigt wird, ist sie für die normale Funktion des WinSTon nicht erforderlich.

Das Laden der Treiber-Dll sowie der Aufruf von Funktionen daraus wurden im Modul `GemdosPCI.cpp` realisiert. Gemdos-Systemaufrufe mit den Operationscode `0xFF` werden zur weiteren Beabeitung aus dem Modul `Gemdos.cpp` in das Modul `GemdosPCI.cpp` weitergereicht.

2.2 Treiberzugriff unter MC68000-Programmen

WinSTon ist ein ATARI-Emulator und stellt MC68000-Programmen nicht nur eine virtualisierte Hardware, sondern auch virtualisierte Systemdienste, wie z.B. Plattenzugriffe, zur Verfügung. WinSTon erkennt, ob ein aufgerufener Systemdienst vom emulierten Betriebssystem abgearbeitet werden kann oder über eigene Funktionen zur Abbildung auf das Gastsystem PC bearbeitet werden muss. Diese Erkennung ist eigentlich für das Zielbetriebssystem TOS implementiert, nutzt aber Hilfsmittel, z.B. illegale Befehlscodes, die auch für anderweitig verwendet werden können.

In WinSTon V0.53 wird der (illegale) Befehlscode 0x0008 als Kennung eines Gemdos-Aufrufes interpretiert. Je nach Funktionscode des Gemdos-Aufrufes werden die Aufrufparameter in Registern oder auf dem Stack erwartet.

Der Aufruf für Funktionen aus `GemdosPCI.cpp` hat den Funktionscode `0xFF` und erwartet sämtliche Parameter auf dem Stack. Die unterschiedlichen, aufzurufenden Funktionen werden durch Sub-Funktionscode definiert. Auf Assemblerebene sieht ein Aufruf wie folgt aus:

`MOVE xxx, -(A7)`	Parameter für `GemdosPCI`-Aufruf auf den Stack
`MOVE =$yy,-(A7)`	Sub- Funktionscode auf den Stack
`MOVE =$FF,-(A7)`	Funktionscode 0xFF (`GemdosPCI`)auf den Stack
`PEA $`	Vorbereitung des Gemdos-Aufrufs
`MOVE =$2700,-(A7)`	Vorbereitung des Gemdos-Aufrufs
`$0008`	Gemdos-Aufruf

Die Auswertung der übergebenen Parameter sowie die Durchführung der entsprechenden Aktion erfolgt im WinSTon, nach Abarbeitung läuft das Programm mit dem folgenden Befehl weiter.

3 Unterstützte Funktionen

Die Funktionen des Treibers werden aus dem WinSTon heraus entsprechend dem Subfunktioncode beim Gemdos-Call 0xFF aufgerufen.

Das Modul `KlibDrv.A68` stellt für jede dieser Funktionen einen PEARL-Funktionsaufruf in Form eines MC68000-Assembler-Unterprogramms bereit. Das Modul `Klib.P` stellt einige vereinfachende Wrapper für normalerweise erforderliche Funktionalitäten bereit.

Eine genaue Beschreibung der einzelnen Funktionsprototypen findet sich in der Datei `KlibDrv.H` bzw. `Klib.H`. Diese Beschreibung soll hier nicht wiederholt werden.

Die unterstützten Funktionen lassen sich in 4 Gruppen gliedern:

3.1 Verwaltung der Treiber-Dll

Die Funktionen `LoadDrv` und `FreeDrv` weisen den WinSTon an, eine namentlich anzugebende Dll zu laden bzw. zu entladen. Jedem Aufruf von `LoadDrv` muss ein Aufruf von `FreeDrv` entsprechen.

Diese Funktionen entsprechen den Windows-API-Aufrufen LoadLibrary bzw. FreeLibrary und sind unabhängig von der Art des eingesetzten Treibers.

Ist das Laden der Dll erfolgreich, so ermittelt der WinSTon die für die weiteren Funktionen erforderlichen Funktionsadressen innerhalb der Dll. Bietet die geladene Dll nicht die erwarteten Funktionen an, so entlädt WinSTon die Dll wieder und kehrt mit der Fehlernummer 0 aus `LoadDrv` zurück.

3.2 Starten/Stoppen des Treibers

Mit den Funktionen `OpenDrv` und `CloseDrv` wird der (vorher geladene!) Treiber gestartet bzw. angehalten. Jedem Aufruf von `LoadDrv` muss ein Aufruf von `FreeDrv` entsprechen. Das Starten des Treibers ist Vorraussetzung für die Nutzung der weiteren Funktionen.

Das Modul `Klib.P` bietet die Wrapperfunktionen `Start_PCI_Driver` und `Stop_PCI_Driver`, die jeweils der Kombination von `LoadDrv` und `OpenDrv` bzw. `CloseDrv` und `FreeDrv` entsprechen und damit deren einzelnen Aufruf erübrigen.

3.3 PCI-Informationsabfrage

Mit der Funktion `GetLastPciBus` kann die Nummer des letzten PCI-Busses im PC erfragt werden. Diese Abfrage ist Voraussetzung für das Suchen auf dem PCI-Bus nach einer bestimmten PCI-Karte.

Mit der Funktion `GetPciDeviceInfo` können die Konfigurationsdaten aller auf den lokalen PCI-Bussystemen vorhandenen Karten ausgelesen werden.

Für die gezielte Suche nach Karten mit einer vorgegebenen Vendor- und Device-ID stehen die Funktionen `FindFirstPCIBase` und `FindNextPCIBase` im Modul `Klib.P` zur Verfügung. Diese Funktionen geben die Basisadresse einer ggf. gefunden Karte zurück.

3.4 Ein-/Ausgabe

Mit den Funktionen `GetPortByte`, `GetPortWord` und `GetPortLong` werden die entsprechenden Datentypen von einer angegebenen Adresse des PCI-Bus gelesen, mit den `Funktionen SetPortByte/Word/Long` werden die entsprechenden Daten ausgeben.

Die tatsächlichen Ein-/Ausgaben erfolgen über den Windows-Treiber und unterliegen nicht mehr der Zugriffskontrolle des Betriebssystems. Bei der Benutzung dieser Funktionen ist entsprechend mit Vorsicht zu verfahren – das Ansprechen von anderen als wohlbekannten Adressen der gewünschten Zielhardware kann undefinierte Folgen haben. Nicht nur der Absturz des Windows, sondern auch der Verlust der Windows-Installation kann eine Folge von Zugriffen auf andere als die bekannten bzw. über `FindxxxPCIBase` erfragten Adressen sein!

4 Installation

Auf der Windows-Seite muss der Treiber für die anzusprechenden Karten gemäss den Vorschriften des Treiberanbieters installiert werden. Erst nach erfolgreicher Installation ist der erfolgreiche Aufruf der Treiberfunktionen möglich; ohne Installation kann zwar ggf. die Treiber-Dll geladen werden, diese kann dann jedoch nicht den tatsächlichen Treiber starten.

Auf der PEARL-Seite ist keine Installation erforderlich. Die entsprechenden Module (`KlibDrv.A68` und `Klib.P`) müssen übersetzt und zusammen mit der Applikation geladen werden.

Der zu dieser PCI-Unterstützung gehörende WinSTon wird mit einem Verzeichnis `winston-rtos-79b-15` ausgeliefert, das bei der Konfiguration des WinSTon (Menue Settings, Reiter Hard Discs) als Hard Disc Directory einzustellen ist und dann für RTOS-UH als Laufwerk `/X0/` zur Verfügung steht.

Die Quelltexte der PEARL-Unterstützung für PCI-Zugriffe finden sich dann im Verzeichnis `/X0/demo`.

5 Ansprache anderer bzw. weiterer Treiber

Die Unterstützung anderer Treiber als des hier beschriebenen Klibdrv der Fa. Kolter electronic erfordert Änderungen sowohl im WinSTon, insbesondere im Modul `GemdosPCI.cpp`, als auch in den entsprechend unterstützenden Funktionen der KlibDrv.A68.

Der einfachste Weg zu Erweiterungen ist der entsprechende Ausbau der Subfunktionscodes. Die jeweilig vorhanden Funktionen können als Vorbild für Erweiterungen dienen. Die Auswertung der Subfunktionscodes findet sich in der Funktion `GemDOS_PCI`.