COMPUTERGESTÜTZTES EXPERIMENTIEREN I

P R A K T I K U M

Kurzbeschreibung der LabPC$^+$-Schnittstellenkarte

Die LabPC$^+$-Schnittstellenkarte der Firma National Instruments vereinigt in sich alle wichtigen Funktionen, die für computergestützte Experimente notwendig sind aber nicht über Standard-Schnittstellen durchgeführt werden können. Dies sind Erfassen und Erzeugen von digitalen Signalen, Erfassen und Erzeugen von analogen Signalen sowie die Implementaion von Echtzeituhren mittels programmierbaren Zählern. In den Praktika CGE I und II werden wir die letztere Option nicht verwenden.

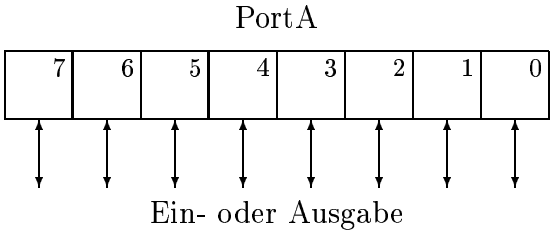
Die LabPC$^+$-Karte wurde ursprünglich für IBM-PCs mit dem ISA-Bus entwickelt. Da die AlphaStation u.a. über einen ISA-Bus verfügt, können die relativ billigen Schnittstellen des PC-Marktes verwendet werden. Der Nachtteil ist, dass der ISA-Bus Byte-orientiert ist, während die Alpha ein 64-Bit-Prozessor ist. Dies hat zur Folge, dass alle Register, die zur Steuerung der Funktionen der LabPC$^+$-Karte dienen, 8 Bit aufweisen. Im Folgenden werden die für das Praktikum relevanten Register und Funktionen beschrieben. Diese Register lassen sich in vier Gruppen zusammenfassen:

1. Konfiguration und Status
2. Analog-Eingabe
3. Analog-Ausgabe
4. digitale Ein-/Ausgabe.

Bei Verwendung des vorgefertigten gemeinsamen Datenbereiches IOSPACE und des Unterprogrammes MAP_SECTION werden die Variablen mit den hier verwendeten Namen auf die richtigen Hardware-Adressen abgebildet. Eine Kopie des Datenbereiches IOSPACE befindet sich in der Datei [cge.beispiele.isa]common.for. Das Unterprogramm MAP_SECTION wird automatisch zu den Programmen "gelinkt", wenn die Programme mit dem Befehl "Experiment" übersetzt und gelinkt werden.

1. Digitale Ein-/Ausgabe

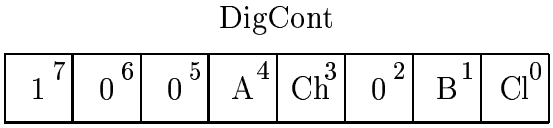
Hierfür stellt die Schnittstelle drei Ports zur Verfügung, die sowohl für die Ein- als auch für die Ausgabe konfiguriert werden können. Jedes Port wird mit einem 8-Bit-Register gesteuert; z. B. PortA:



wobei die einzelnen Bits über separate Leitungen aus dem Computer herausgeführt werden.

Die Register werden mit den Variablen PortA.REG, PortB.REG und PortC.REG angesprochen. Als Besonderheit ist anzumerken, dass die niederwertige und die höherwertige Hälfte (4 Bit = 1 Nibble) von PortC separat für die Ein-/Ausgabe konfiguriert werden können.

Das Konfigurieren dieser Register erfolgt mit dem Register DigCont,



wobei die mit Buchstaben bezeichneten Bits bestimmen, ob die entsprechenden Ports (A ganzes PortA, B ganzes PortB, Cl niederwertiges Nibble von PortC, Ch höchstwertiges Nibble von PortC) für die Ausgabe (0) oder für die Eingabe (1) konfiguriert werden sollen. Z.B. konfiguriert die Zuweisung DigCont.REG='89'x PortA und PortB für die Ausgabe und das ganze PortC für die Eingabe. Eine nachfolgende Zuweisung PortA.REG = 5 setzt die Leitungen, die mit Bit 0 und mit Bit 2 von PortA verbunden sind, auf +5 V und die übrigen Leitungen auf 0 V.

2. Analoge Ausgabe

Die analoge Ausgabe ist ebenso einfach wie die digitale Ausgabe. Für die Ausgabe stehen zwei Digital/Analog-Wandler (DAC0, DAC1) zur Verügung, mit welchen Spannungen zwischen -5 V und +5 V erzeugt werden können. Die Zuweisung eines Zahlenwertes zu den zugehörigen Registern führt zur Erzeugung der entsprechenden Spannung der analogen Ausgänge. Da die D/A-Wandler über eine Auflösung von 12 Bit verfügen, benötigt man 2 Bytes, und deshalb 2 Register, um den digitalen Wert zu speichern. Das sind DAC0L für die niederwertigen 8 Bits und DAC0H für die restlichen höherwertigen Bits von DAC0; DAC1L und DAC1H sind die entsprechenden Register von DAC1, z.B.:

DAC0H								DAC0L							
x^7	x^6	x^5	x^4	11^3	10^2	9^1	8^0	7^7	6^6	5^5	4^4	3^3	2^2	1^1	0^0

Die digitalen Werte sind "straight binary" kodiert; d.h. 0 entspricht -5 V, 2048 entspricht 0 V und der maximale Wert $2^{12} - 1 = 4097$ entspricht 4.9976 V. Die Auflösung des A/D-Wandlers beträgt also 10 V/2^{12} = 2.44 mV. Beispielsweise erzeugt der Wert $2458_{10} = 99A_{16}$ eine Spannung von +1.00 V. Will man 1.00 V mit DAC0 ausgeben, muss man also DAC0L.REG = '9A'x und DAC0H.REG = '9'x zuweisen.

3. Analoge Eingabe

Die analoge Eingabe ist die komplexeste, hier beschriebene Operation, deren Durchführung einer ganzen Reihe von Registern bedarf. Um unter klaren Bedingungen zu Arbeiten, empfiehlt es sich daher, die LabPC$^+$-Karte zu initialisieren. Dafür steht das vorgefertigte Unterprogramm CLR_LAB_PC zur Verügung, welches mit "Experiment" automatisch zu den Programmen gelinkt wird.

Die LabPC$^+$-Karte ermöglicht in Kombination mit den Echtzeituhren verschiedene Betriebsoptionen. Im Praktikum wenden wir nur die einfachste Option, das Erfassen einzelner analoger Werte, an, bei der keine Echtzeituhr benutzt wird. Diese Operation gliedert sich in drei Schritte:

1. Konfigurieren des A/D-Wandlers
2. Initiieren der A/D-Umsetzung und Abwarten deren Beendigung
3. Lesen des Ergebnisses der A/D-Umsetzung.

3.1 Konfigurieren des A/D-Wandlers

Dem A/D-Wandler vorgelagert befindet sich ein analoger Auswahlschalter (Multiplexer), mit dem in der differentiellen Betriebsart (s.u.) einer von 4 analogen Eingangskanälen (0-3) ausgewählt werden kann. Mit den Bits K1 und K2 im Register ComReg1 wird der Eingangskanal festgelegt, wobei <K2 K1> eine zweistellige Dualzahl bilden, die die Darstellung der Zahlen 0 bis 3 erlaubt.

ComReg1

0^7	$V2^6$	$V1^5$	$V0^4$	0^3	$K2^2$	$K1^1$	0^0

Des weiteren ist dem A/D-Wandler ein programmierbarer Vorverstärker vorgeschaltet, um die Auflösung des Wandlers optimal ausnutzen zu können. Die Kodierung der Verstärkung im Bereich von 1 bis 100 erfolgt mit den Bits V0 bis V2 in ComReg1 gemäss folgender Tabelle:

V2V1V0	Verstärkung
000	1
001	1.25
010	2
011	5
100	10
101	20
110	50
111	100

Die übrigen Bits von ComReg1 müssen für unsere Anwendung 0 gesetzt werden. Für den einfachsten Fall, Kanal 0 und Verstärkung 1, genügt es also, ComReg1.REG = 0 zu setzten.

Bei weit entfernten Signalquellen ist es ratsam, die Spannung erdfrei zu messen, d. h. es wird die Spannungsdifferenz zwischen den beiden Signalleitungen gemessen. Diese diffentielle Messmethode wählt man aus, indem man Bit3 des Registers ComReg4 setzt: ComReg4.REG = 8.

3.2 Initiieren der A/D-Umsetzung und Abwarten deren Beendigung

Die A/D-Wandlung muss explizit gestartet werden, was durch Schreiben in das Register

StartConv geschieht; z.B.: StartConv.REG = 1. Die A/D-Umsetzung benötigt etwa 12 μs. Die Beendigung der Umsetzung wird durch eine Eins im Bit0 des Statusregisters angezeigt. Da die Dauer der Umsetzung nicht genau vorhersagbar ist, muss das Ende durch wiederholte Statusabfrage festgestellt werden. Das Statusregister weist dieselbe Hardware-Adresse auf wie ComReg1. Der Inhalt des Statusregisters wird angezeigt, wenn lesend auf ComReg1 zugegriffen wird. Schreibender Zugriff hingegen setzt, wie oben beschrieben, den Inhalt von ComReg1; ComReg1 hat also zwei Funktionen!

3.3 Lesen des Ergebnisses der A/D-Umsetzung

Das Ergebnis der A/D-Wandlung wird wie bei den D/A-Wandlern mit 12 Bit in der "straight binary" Kodierung (s.o.) dargestellt. Es wird in einem Zwischenspeicher aufgenommen, der insgesamt 512 Bytes speichern kann, ein sogenannter Fifo-Speicher. Da dieser Speicher Byte-orientiert ist, wird das Ergebnis der Wandlung in zwei aufeinanderfolgenden Bytes gespeichert. Durch Lesen des Registers ADFifo.REG wird zuerst das niederwertige Byte, durch wiederholtes Lesen das höherwertige, zurückgegeben. Wie bei der D/A-Wandlung kann dann mit diesen beiden Bytes und unter Berücksichtigung der Verstärkung die Spannung des Signals in Volt berechnet werden.