Fachhochschule Kaiserslautern Prof. Dr. P. Liell
Echtzeitbetriebssysteme/ Prozessdatenverarbeitung II
SS 2012 1. Versuch

Allgemeines zum Ablauf und zur Versuchsausarbeitung

Es ist eine Ausarbeitung zum Labor zu erstellen, die jede Aufgabe und jeden Versuch behandelt. Alle gestellten Fragen sind dort zu beantworten und die Lösungen der Aufgaben zu beschreiben. Erstellte Programme sind verständlich zu dokumentieren. Die Dokumentation ist so knapp wie möglich, aber so ausführlich wie nötig zu halten. Die erstellten Programme sind als kommentierter Quelltext und als lauffähiges Programm auf der Diskette abzuspeichern. Die Diskette muss mit der Ausarbeitung abgegeben werden.

Es ist zum festgelegten Termin je Gruppe eine Ausarbeitung abzugeben, die alle Versuche umfasst. Die Einteilung der Gruppen ist fest; die Namen aller Gruppenmitglieder sind auf der Ausarbeitung anzugeben. Sollte es Probleme mit der Zusammenarbeit geben, ist **unverzüglich** der Betreuer zu benachrichtigen.

Unabhängig von der Ausarbeitung muss jederzeit mit Befragungen zu jedem Versuch und jedem Programm gerechnet werden. Außerdem muss innerhalb eines Testates jeder Teilnehmer zur Notenfindung selbständig ein kurzes Programm erstellen.

Weitere Unterlagen zum Labor

Die aktuelle Aufgabenstellung und weitere Dokumentationen finden Sie in den Ordnern im Schrank und auf der Internetseite zum Labor:

www.fh-kl.de/~peter.liell -> Vorlesungen -> Labor Echtzeitbetriebssysteme -> Internes für Laborteilnehmer

Inbetriebnahme des Systems

Die Systeme bestehen aus der FORCE-VME-Bus-Rechnereinheit (Target) und zwei Terminals. Als Terminals werden WINDOWS-Rechner verwendet, die mit Hilfe des Terminalprogramms HYPERTERMINAL zu VT52-Terminals umfunktioniert werden. Die Rechner arbeiten im Studentenmodus ohne Passwort.

Schalten Sie zuerst die PCs ein! Erst anschließend sollten die FORCE-Systeme am Schlüsselschalter bzw. am Schalter auf der Rückseite des Gerätes eingeschaltet werden, denn nur dann ist die erste Meldung des Systems am Bildschirm sichtbar.

Es meldet sich nach dem Hochlauf das Echtzeitbetriebssystem VMEPROM. Es können nun VMEPROM-Kommandos (build in commands) eingegeben werden. Diese kennt (hoffentlich) Ihr *Spezialist*. Eine Dokumentation aller Befehle finden Sie im Ordner und auf der Internetseite zur Veranstaltung.

Verwenden Sie bei allen Arbeiten am VME-System nur die 5 1/4"-Diskette, nicht die Festplatte; verwenden Sie keine eigenen Disketten!. Die Laufwerksbezeichnung ist *F0* (Großbuchstabe!), manchmal genügt *0*.

Der Drucker ist am PC angeschlossen und kann daher nicht über einen VMEPROM-Befehl zum Ausdrucken einer Datei verwendet werden. Es kann daher nur vom Hyperterminal aus gedruckt werden.

Die Systeme können zu jedem Zeitpunkt direkt am Schlüsselschalter bzw. am Schalter auf der Rückseite des Gerätes ausgeschaltet werden. Sie müssen, im Gegensatz zu UNIX, nicht "heruntergefahren" werden. Versuchen Sie daher auch nicht, UNIX zu laden!

Vergessen Sie beim Ausschalten den PC und dessen Bildschirm nicht!

1. Aufgabe

Formatieren Sie eine Diskette.

Verwenden Sie den *Format*-Befehl mit den Standardeinstellungen.

Versuchen Sie, die Diskette anzusprechen (z.B. über *Inhaltsverzeichnis ausgeben*). Sie werden feststellen. dass weitere Aktionen notwendig sind, eine Diskette für das System nutzbar zu machen. Finden Sie diese heraus!

2. Aufgabe

Nehmen Sie den *Line-Assembler* in Betrieb.

Schreiben Sie einige Zeilen Assembler-Code mit Hilfe des *Line-Assemblers* direkt in den Speicher. Verfügbar ist der Speicherbereich ab Adresse $8000. Speichern Sie das Programm auf Diskette ab. Laden Sie es wieder in den Speicher und starten Sie es. Testen Sie die Funktionen *Trace, Registerinhalte anzeigen, Speicherinhalt anzeigen, Disassemblieren* usw.

3. Aufgabe

Schreiben Sie ein Progammstück im *Editor*. Speichern Sie es auf Diskette ab und versuchen Sie dann, diese Datei mit dem *Line-Assembler* zu assemblieren.

Wenn Sie glauben, Ihr Terminal spinnt nach dem Aufruf des Editors, hilft die Tastenfolge CTRL-C, CTRL-C und danach (hoffentlich) eine Nachfrage beim *Spezialisten*, was Sie falsch gemacht haben!

4. Aufgabe

Testen Sie auch einmal den Aufruf eines Systemaufrufes Ihrer Wahl (Auch dafür gibt es einen Spezialisten!). Prüfen Sie, wie Fehler bei der Ausführung des Systemaufrufs zurückgemeldet werden. **Wo steht die Fehlernummer? Beachten Sie, dass sich in dem Register, das die Fehlernummer enthält, auch im Nicht-Fehlerfall einen Wert befindet. Wodurch wird angezeigt, dass es sich überhaupt um einen Fehler handelt?** Überprüfen Sie dies, indem Sie den gleichen Systemaufruf einmal mit und einmal ohne Fehlermeldung testen.

Fachhochschule Kaiserslautern	Prof. Dr. P. Liell
Echtzeitbetriebssysteme	
SS 2012	2. Versuch

1. Aufgabe

Programmieren Sie den *Peripheral Interface Timer* (PIT) 68 230 auf der ELTEC-Erweiterungsplatine.

Stellen Sie zunächst die physikalische Adresse eines der beiden PITs fest. In der Beschreibung zur Erweiterungskarte stehen alle notwendigen Informationen hierzu. Schauen Sie im Zweifelsfall in den Schaltbildern nach!

Beachten Sie die Belegung des *Address Modifier*! Dies sind 6 zusätzliche Leitungen zum Adressbus, die von der CPU generiert und von der Erweiterungskarte ausgewertet werden. Die Erzeugung und die hierzu notwendige Softwareeinstellung müssen Sie mit den Spezialisten für die CPU-Karte abklären, die Spezialisten für die Erweiterungskarte sollten die notwendige Belegung kennen.

Versuchen Sie, eine LED auf der Frontseite der Erweiterungskarte definiert einzuschalten. Schalten Sie hierzu den PIT auf Ausgabe (Mode 0). Wenn die Programmierung der Register fehlerfrei gelingt, die LEDs aber nicht das gewünschte Muster zeigen, sollten Sie im Schaltplan nachschauen, was die Ursache sein könnte. Das Gleiche gilt, wenn die Fehlermeldung *spurious* auftritt (= undefinierter Interrupt).

2. Aufgabe

Programmieren Sie ein Lauflicht.

Es soll ein Binärmuster mit einer vorgegebenen Frequenz rotierend auf den LEDs A0 bis B7 von Port 1 angezeigt werden. Achten Sie darauf, dass die Basisdressen der Ports in Registern abgelegt werden, die Register des PITs also indiziert adressiert werden.

Überlegen Sie sich verschiedene Möglichkeiten der Erzeugung der Verzögerungszeit und diskutieren Sie die Vor- und Nachteile. Realisieren Sie Ihr Programm unter Verwendung der von Echtzeitbetriebssystem vorgegebenen zeitabhängigen Eventsteuerung.

Legen Sie zunächst Muster und Frequenz nur in Registern ab. Erweitern Sie Ihr Programm dann so, dass Muster und Frequenz an einer Speicheradresse im Variablenbereich abgelegt sind, der hinter Ihrem Programm im Speicher liegt. Achten Sie darauf, dass beim Starten des Programms als Task die Adresse, an der Ihr Programm abläuft, vom Betriebssystem vorgegeben wird und damit zunächst nicht bekannt ist!

Speichern Sie das Programm nach dem Test auf Diskette ab und starten Sie es als Task mit dem entsprechenden VMEPROM-Kommando.

3. Aufgabe

Erstellen Sie eine 2. Task, die die gleiche Aufgabe für A0 bis B7 von Port 2 erfüllt mit einer Frequenz, die nur wenig von der ersten abweicht. Starten Sie dieses Programm ebenfalls als Task und prüfen Sie die Auswirkungen von Änderungen der Prioritäten.

4. Aufgabe

Lösen Sie die Aufgabenstellung von Aufgabe 2 interruptgesteuert unter Verwendung des Timerbausteins 68230 auf der Erweiterungsplatine!

Fachhochschule Kaiserslautern	Prof. Dr. P. Liell
Echtzeitbetriebssysteme	
SS 2012	3. Versuch

1. Aufgabe

Starten Sie eine Task VMEPROM, die Port 2 als Ein- und Ausgabeport benutzt. Wenn Sie am 2. PC, der an Ihrem Rechner angeschlossen ist, ebenfalls die Terminalemulation starten, können Sie dann 2 Projekte am gleichen Rechner bearbeiten.

2. Aufgabe

Es soll ein Parkscheinautomat mit Hilfe von 2 Tasks realisiert werden.

Der Parkscheinautomat soll nach Einwurf von 1 Euro einen Parkschein ausgeben. Es können 10 C, 50 C und 1 Euro in beliebiger Reihenfolge eingeworfen werden. Sobald der notwendige Betrag erreicht ist, soll ein Parkschein und ggf. das Wechselgeld ausgegeben werden. Eine Task soll den Münzprüfer realisieren, die zweite Task den eigentlichen Automaten

Zur Kommunikation zwischen Münzprüfer und Automat sollen folgende Signale dienen, die zwingend vorgeschrieben sind:

EINW = 1_{10}= 1_2:	von Münzprüfer an Automat	es wurden 10 C eingeworfen
EINW = 2_{10}= 10_2:	von Münzprüfer an Automat	es wurden 50 C eingeworfen
EINW = 3_{10}= 11_2:	von Münzprüfer an Automat	es wurden 1 € eingeworfen
EINW = 0_{10}= 0_2:	von Münzprüfer an Automat	Programm beenden

Die Kommunikation soll über die Systemaufrufe *Set Message Pointer* (XSMP) und *Get Message Pointer* (XGMP) erfolgen. Verwenden Sie der Einfachheit halber den übergebenen Zeiger nicht als Adresse der Nachricht, sondern als Nachricht selbst. Die oben benannten Signale sollen Bits in dieser Nachricht darstellen.

Teilen Sie nach Festlegung der Kommunikation Ihre Gruppe auf und bearbeiten Sie die beiden Teilprojekte *Münzprüfer* und *Automat* an getrennten Terminals.

Münzprüfer:

Der Münzprüfer liest von der Tastatur des Terminals 2 die Ziffernfolgen 0, 10, 50 und 100 mit anschießender Eingabetaste ein. Sie stellen die eingeworfenen Beträge von 10 C, 50 C und 1 € dar. 0 stellt Programmabbruch dar. Ungültige Werte werden nicht akzeptiert. Nach Drücken der Eingabetaste sollen nach jedem Einwurf die oben angegebenen Ausgabesignale erzeugt werden.

Automat:

Der Automat erhält vom Münzprüfer nach jedem Einwurf die oben angegebenen Signale. Er überprüft, ob mindestens 1 € erreicht ist und gibt dann einen Parkschein aus. Dies soll durch Aufleuchten einer grünen LED für 2 Sekunden angezeigt werden. Außerdem soll für 2 Sekunden der zurückzuzahlende Betrag (Wechselgeld) durch rote LEDs angezeigt werden. 1 LED bedeutet 10 C, 2 LEDs bedeuten 20 C usw.

Aufgabe:

Legen Sie die Kommunikation zwischen den Tasks fest und bestimmen Sie den verwendeten Port. **Zeichnen Sie zunächst einen Zustandsgrafen für den Automaten, dann jeweils einen Programmablaufplan für Münzprüfer und Automaten und beginnen Sie erst dann mit der eigentlichen Programmierung!**