

**INSTITUT FÜR INFORMATIK**  
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN



Informatik VI: Robotics and Embedded Systems  
Prof. Dr. A. Knoll

## **Prozessrechner-Praktikum Echtzeitsysteme**

**Einführung in VxWorks**

**C. Buckl**                      **G. Schrott**  
[buckl@in.tum.de](mailto:buckl@in.tum.de)    [schrott@in.tum.de](mailto:schrott@in.tum.de)

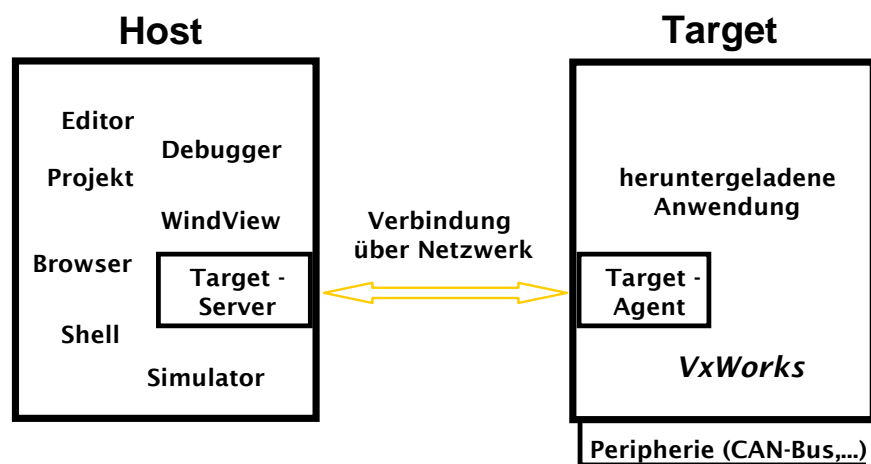
Sommersemester 2006



## 1 Einführung zu VxWorks

Im Praktikum "Echtzeitsysteme" wird das Echtzeitbetriebssystem VxWorks eingesetzt. VxWorks wurde von der Firma Wind River Systems ([www.windriver.com](http://www.windriver.com)) entwickelt. Das Betriebssystem ist für den Entwurf verteilter, zeitkritischer Anwendungen ausgelegt und bietet aus diesem Grund nur eine Eingabemöglichkeiten über die Kommandozeile und keine graphische Oberfläche. Um dennoch eine komfortable Entwicklung zu gewährleisten, erfolgt die Applikationsentwicklung auf einem Entwicklungsrechner (Host) mit einem Standardbetriebssystem (in unserem Fall Windows) und dem Werkzeug Tornado. Entwickelte Anwendungen können mit Hilfe dieses Werkzeugs auf den eigentlichen Zielrechner (Target) herunter geladen und dort ausgeführt werden. Zur Steuerung der Programmausführung ist eine so genannte RemoteShell in Tornado integriert. Selbstverständlich ist auch eine reine Ausführung auf dem Zielrechner möglich, jedoch für unser Praktikum nicht sinnvoll.

Auf dem Hostrechner kann die Entwicklung der Echtzeitapplikationen (Codierung, Kompilieren, Debugging, Simulation, ...) komfortabel durchgeführt werden. Dazu wird mit Tornado eine umfangreiche Entwicklungsumgebung bereitgestellt. Über sie lässt sich Analyse-Software wie der Debugger, der Browser und WindView bedienen. Zum Testen der entwickelten Applikationen bieten sich dem Entwickler zwei Möglichkeiten: die Simulation der Anwendung im VxWorks-Simulator (ausgeführt auf dem Entwicklungsrechner) oder die Ausführung der Anwendung auf dem Zielrechner. Ein Nachteil beim Testen mit dem Simulator besteht darin, dass externe Ereignisse, z.B. Interruptanforderungen durch Peripheriekomponenten, simuliert werden müssen. Es ist also nicht ohne weiteres möglich, mit dem Simulator die volle Funktionsfähigkeit jedes beliebigen Realzeitprogramms zu testen. Überträgt man hingegen das zu testende Programm auf einen Zielrechner, so steht die gesamte Hardware des Zielrechners mit all seinen Peripheriekomponenten zur Verfügung. Somit kann das Programm ohne größeren Aufwand getestet werden. Abbildung 1 gibt einen groben Überblick über verschiedene, wichtige Komponenten des Tornado-Entwicklungssystems, auf die in den folgenden Kapiteln des öfteren Bezug genommen wird.



**Abb. 1** Grobübersicht über einige Komponenten von Tornado und die Beziehung zwischen Host und Target

Ein Entwicklungsrechner kann mit mehreren Zielrechnern verbunden sein. Mit Hilfe der Tornado-IDE und von Shells, die auf dem Entwicklungsrechner oder den Zielrechnern gestartet werden, kann ein Benutzer interaktiv Einfluss auf die echtzeit-kritische VxWorks-Anwendung nehmen. Beispielsweise können durch Kommandos, die über Shells abgesetzt werden, Anwendungen vom Ent-

wicklungs- auf einen Zielrechner übertragen, dort ausgeführt und auch wieder vom Zielrechner gelöscht werden. Des Weiteren ist es möglich, Komponenten des zu testenden Programms zu überwachen, um die gewonnenen Daten für spätere Auswertungen, z.B. für die Fehlersuche, heranzuziehen. Für die Kommunikation zwischen Entwicklungs- und Zielrechner sind der Target-Server und der Target-Agent zuständig. Der Target-Agent arbeitet unabhängig von der Applikation und von VxWorks.

Das Herz des Laufzeitsystems von VxWorks ist der „wind“-Microkernel. Er ist skalierbar; man kann also die Komponenten des Kernels je nach Bedarf zusammenstellen. Der Kernel stellt Funktionen wie Speichermanagement, eine Multitasking-Umgebung, Zeitdienste, Interprozesskommunikations- und Synchronisations-Mechanismen zur Verfügung. Der Kernel ist immer Bestandteil eines VxWorks-Images, das nach dem Booten eines Zielrechners von Diskette automatisch über das Netzwerk auf den Rechner übertragen wird. Das VxWorks-Image liegt auf den Rechner *atknoll41*. Für alle Zielrechner wird dasselbe Image verwendet. Es ist entsprechend der Aufgabenstellung so konfiguriert, dass die Funktionalität aller Zielrechner angesprochen werden kann. Spezielle Treiber, die ab Aufgabe 4 einzubinden sind, werden mit den Aufgabenstellungen verteilt. Eine Änderung des Images ist im Rahmen des Praktikums nicht nötig.

Alle Handbücher zu Tornado und VxWorks finden Sie im Verzeichnis *C:\Tornado\Docs* auf Ihrem Hostrechner. Die Handbücher können über die Tornado-IDE durch Auswahl des Menüpunkts *Help* -> *Manual Contents* aufgerufen werden.

## 2 Tornado

Wie bereits erwähnt, umfasst Tornado neben VxWorks weitere Komponenten. Auf einige von ihnen soll in diesem Kapitel näher eingegangen werden. Tornado setzt sich zusammen aus:

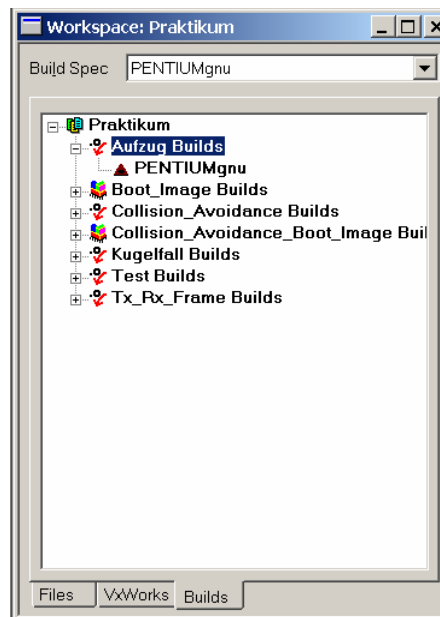
- einem integrierten *Editor*
- einer *Projektverwaltung*
- einem integrierten *C* und *C++ Compiler* sowie *make* (alles GNU)
- einem *Browser*, der aus einer Sammlung von Visualisierungshilfen besteht und mit dem Vorgänge im Zielrechner beobachtet werden können
- *CrossWind*, einem graphisch erweiterten Debugger
- *WindSh*, einer Kommando-Shell mit der Prozesse auf dem Zielrechner gestartet und C-Kommandos zur Ausführung abgesetzt werden können
- einem Simulator, *VxSim*, mit dem ein Zielrechner unter gewissen Einschränkungen simuliert werden kann
- einer integrierten Version von *WindView*, einem Software-Logik-Analysator zum detaillierten Beobachten von Vorgängen im Target
- sowie aus vielen weiteren, teils kundenspezifischen Werkzeugen (speziellen Editoren, CM-Tools, ...)

## 2.1 Projekte

Alle Arbeiten zur Erstellung von Anwendungen in VxWorks finden im Kontext von Projekten statt. Ihre Verwendung und mit Projekten verbundene Begriffe werden in diesem Abschnitt erläutert.

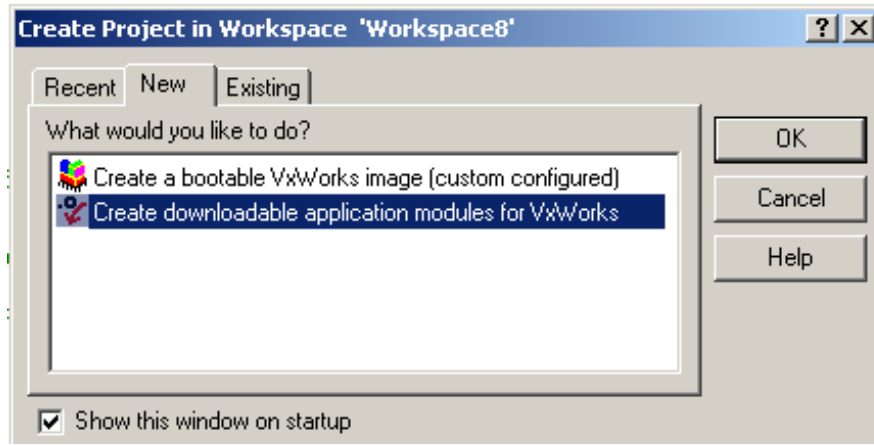
### 2.1.1 Arbeitsbereiche und Projekte

Ein Arbeitsbereich (Workspace) ist ein logischer Container für ein oder mehrere Projekte. Man sollte alle Projekte, die inhaltlich miteinander zu tun haben, in einem Arbeitsbereich ablegen. Abbildung 2 zeigt den Arbeitsbereich *Praktikum*, der auf dem Rechner *atknoll41* angelegt wurde. Die einzelnen Projekte heißen *Aufzug*, *Boot\_Image*, usw. .



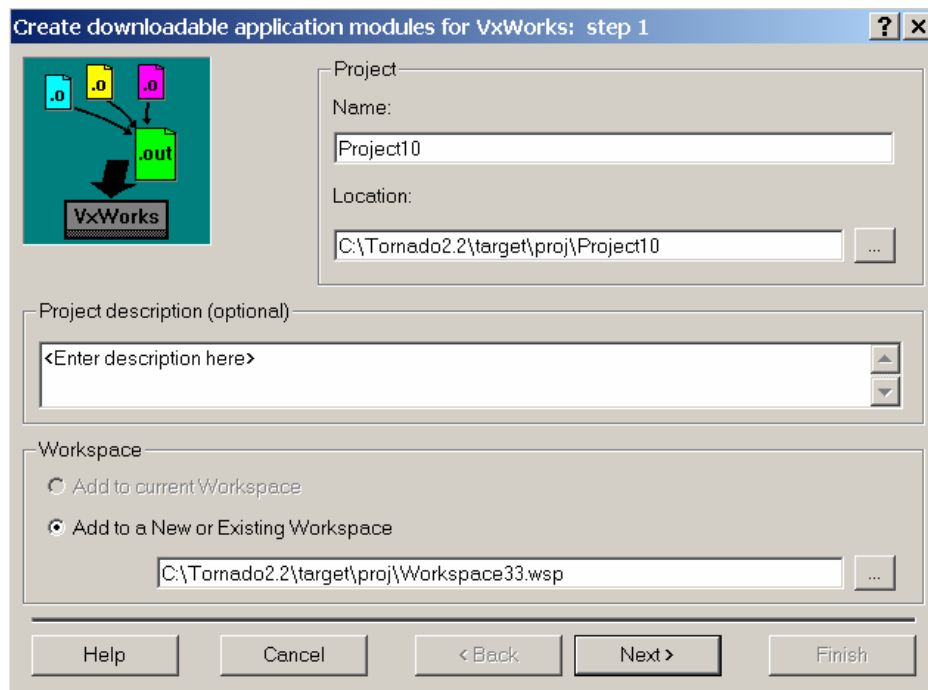
**Abb.2** Workspace *Praktikum* (auf dem Rechner *atknoll41*)

Jede Anwendung für VxWorks wird einem Projekt zugeordnet. Jedes Projekt benötigt sein eigenes Verzeichnis. Dort liegen der Quellcode, der auf mehrere Dateien verteilt sein kann, sowie Informationen zur Verwaltung des Projekts. Um ein neues Projekt ins Leben zu rufen, klickt man auf *File->new Project* und entscheidet anschließend, ob eine herunterladbare Applikation oder bootbares VxWorks-Image erstellt werden soll (vgl. Abb.3). Im Zusammenhang mit den Praktikumsaufgaben dürfen nur herunterladbare Anwendungen erstellt werden und niemals bootbare Images. Ein bootbares Image für alle Rechner liegt bereits auf dem Rechner *atknoll41* vor. Wie bereits erwähnt wird es beim Booten auf den Zielrechner übertragen und Ihre Anwendung kann, nachdem sie ebenfalls auf den Rechner übertragen wurde, auf das Image aufsetzen. Für weitere Erläuterungen im Manual wird nicht mehr zwischen Anwendung und Projekt unterschieden, weil davon ausgegangen wird, dass jedem Projekt nur eine Anwendung zugeordnet ist.



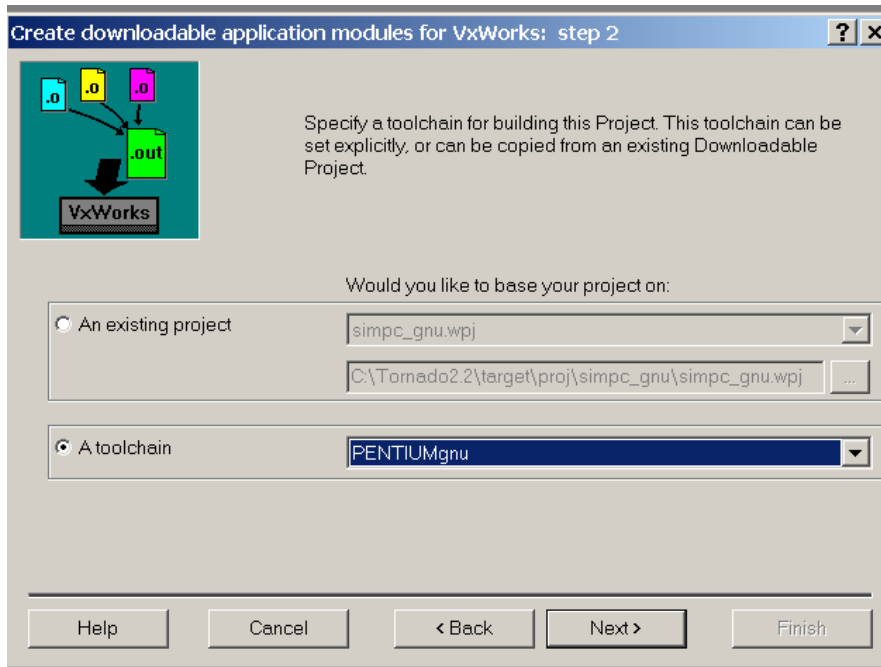
**Abb.3** Projekterstellung: Auswahl zwischen VxWorks-Image und herunterladbarer Anwendung

Im nächsten Schritt gibt man einen Namen, den Speicherort und optional eine Beschreibung zum Projekts an. Es wird auch festgelegt, zu welchem Arbeitsbereich das Projekt hinzugefügt werden soll. Durch die Erstellung eines neuen Projektes kann von der Tornado-IDE automatisch ein neuer Arbeitsbereich vorgeschlagen werden (vgl. Abb.4 – Workspace33.wsp). Man kann das neue Projekt aber auch einem anderen, bereits existierenden Arbeitsbereich hinzufügen, indem dieser über die Schaltfläche [...] gewählt wird.



**Abb.4** Angabe des Projektnamens und Projekt-Workspaces

Nun muss eine Werkzeugkette (Toolchain) für das Projekt (Erklärung zur Werkzeugkette, siehe Abschnitt 2.1.2.) angegeben werden. Ein neues Projekt kann auf den bereits erfolgreich eingesetzten Einstellungen eines älteren Projektes aufbauen. In einem solchen Fall können die Einstellungen vom älteren Projekt einfach übernommen werden. Andererseits kann, wie in Abb.5 zu sehen ist, eine neue Werkzeugkette ausgewählt werden. In Abb.5 wird die PENTIUMgnu-Werkzeugkette für das neue Projekt eingesetzt. Diese muss im Praktikum immer dann eingesetzt werden, wenn ein Zielrechner zur Ausführung des Programms eingesetzt werden. Bei Verwendung des Simulators ist die Simulator-Werkzeugkette (SIMNTgnu) zu benutzen.



**Abb.5** Wahl der Toolchain für ein neues Projekt

### 2.1.2 Werkzeugketten für Zielrechner und den Simulator

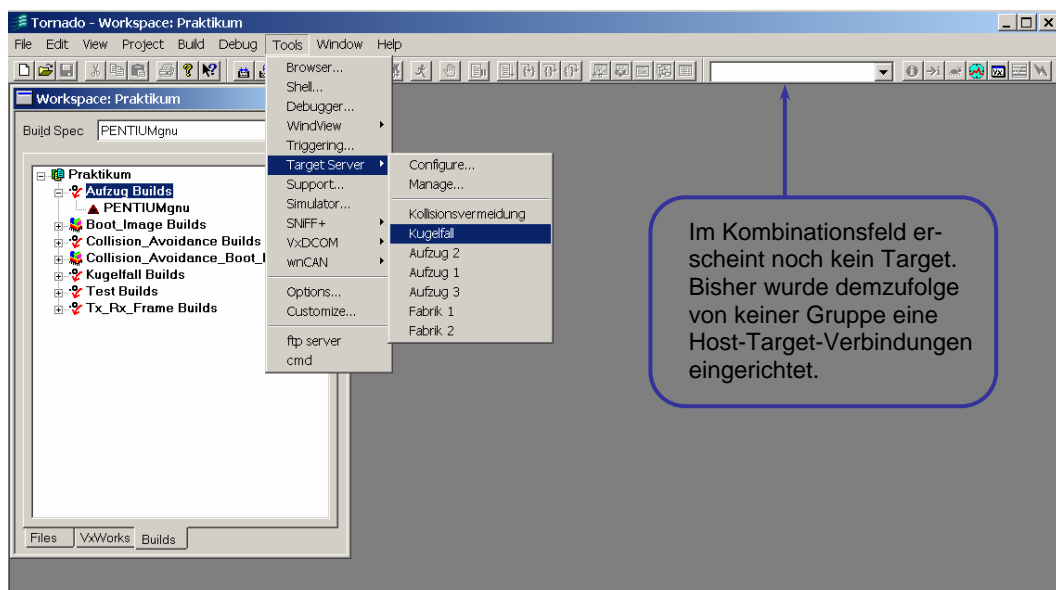
Eine Werkzeugkette ist eine Sammlung von Werkzeugen, die auf bestimmte Hardwarekonfigurationen zugeschnitten sind und erlaubt die Erstellung von Anwendungen für die Plattformen, sowie das Nutzen spezieller Hardwareeigenschaften. Beispielsweise werden durch eine PENTIUM-Werkzeugkette einer Anwendung die speziellen Vorteile, die ein bestimmter Pentium-Prozessor zu bieten hat, zugänglich. Sie brauchen sich nicht darum zu kümmern, wie Teilen der Anwendung der Zugang zu den Eigenschaften der Hardware ermöglicht wird. Diese Aufgabe übernimmt die Werkzeugkette. Ab Aufgabe 4 muss die PENTIUMgnu-Werkzeugkette verwendet werden, weil das VxWorks-Image der Targetrechnern mit ihr übersetzt wurde.

Ein Simulator wird ebenfalls wie eine Zielplattform behandelt, obwohl er, wie der Name schon vermuten lässt, lediglich Hardware simuliert. Der Simulator läuft auf dem jeweiligen Entwicklungsrechner. Durch die SIMNTgnu-Werkzeugkette werden einer Anwendung die speziellen Eigenschaften des Simulators zugänglich gemacht. Weil der Simulator auf dem Entwicklungsrechner läuft, muss er sich die Ressourcen mit anderen Rechenprozessen teilen. Ein Simulator kann sich deshalb nicht in allen Punkten wie ein echter Zielrechner verhalten. Beispielsweise kann es zu Abweichungen bei Zeitmessungen kommen; ein Einfluss, der bei der Anwendungsausführung bzw. beim Testen berücksichtigt werden muss.


### 2.1.3 Host-Target-Verbindungen, das Starten eines Target-Servers und des Simulators

Bevor der Entwicklungsrechner mit einem Zielrechner verbunden werden kann, müssen die im Abschnitt 2.6.2 *Anbinden eines Hosts an die zentrale Tornado-Registry* aufgeführten Schritte durchgeführt sein. Wurden die Schritte schon von einer früheren Praktikumsgruppe durchgeführt, dann müssen sie nicht wiederholt werden. Sie erkennen die erfolgte Durchführung an den eingetragenen Zielrechner im Menü *Tools -> Target Server* (in Abbildung 6 *Kollisionsvermeidung, Kugelfall, Aufzug2, ...*).

Eine Verbindung zu einem Zielrechner sollte nur dann hergestellt werden, wenn dieser nicht schon von einer anderen Gruppe benutzt wird, also keine Verbindung mit dem entsprechenden Rechnernamen im Kombinationsfeld der Tornado-IDE auftaucht. In Abbildung 6 wird eine Verbindung zum Zielrechner *Kugelfall* hergestellt. Durch Auswahl des Menüpunktes *Tools->Target Server* öffnet sich ein Menü, in dem jeder im Praktikumsraum vorhandene Zielrechner erscheint.



**Abb.6** Verbindungsaufbau zum Target *Kugelfall*

Mit Auswahl eines Zielrechners wird auf dem Entwicklungsrechner der Target-Server gestartet, durch den hostseitig die Verbindung zum Zielrechner kontrolliert wird. Einen gestarteten Target-Server erkennt man an einer kleinen roten Zielscheibe  rechts unten in der Windows-Symboleiste.

Vor dem Ausführen der Anwendung muss diese zunächst auf den Zielrechner heruntergeladen werden. Eine Beschreibung zum Herunterladen einer Anwendung finden sie im Kapitel 2.1.4. *Objektmodule auf einen Rechner herunterladen*. Es gibt für das Herunterladen von Objektmodulen wie auch für viele andere Funktionalitäten, die mit Tornado bewerkstelligt werden können, mehrere Möglichkeiten, die nötigen Kommandos abzusetzen. Eine Variante, die für alle Funktionalitäten von Tornado möglich ist, ist das Absetzen der Kommandos über eine Host-Shell. Häufig verwendete Kommandos lassen sich leichter über die Tornado-IDE absetzen, wenngleich dabei nicht unbedingt alle Parameter zu einem Kommando berücksichtigt werden müssen.

Zum Beenden einer Verbindung mit dem Zielrechner wählen Sie im Menü *Tools->Target Server->Manage* im Kombinationsfeld *Targets* den entsprechenden Rechner, dessen Verbindung beendet werden soll. Nachdem das Target blau hinterlegt erscheint, wählen Sie im Kombinationsfeld *Select*



Action den Eintrag *kill* aus und bestätigen mit Klicken auf *Apply*. Nur diese Vorgehensweise garantiert einen sicheren Verbindungsabbau!

Der Simulator wird durch Klicken auf das Symbol  gestartet. Es öffnet sich ein Fenster wie in Abbildung 7.

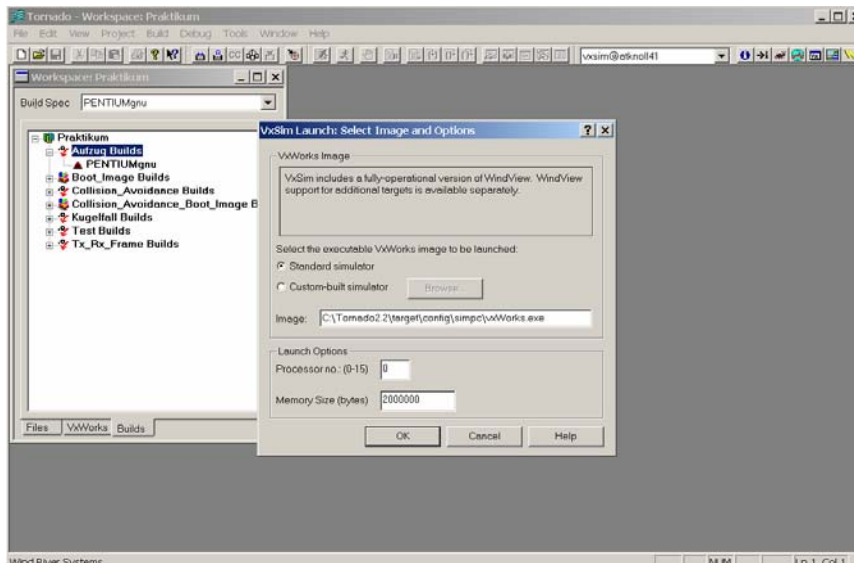


Abb.7 erster Dialog beim Starten des Simulators

Das Fenster sollte mit OK bestätigt werden, genauso wie der nächste Dialog, der in Abbildung 8 zu sehen ist.

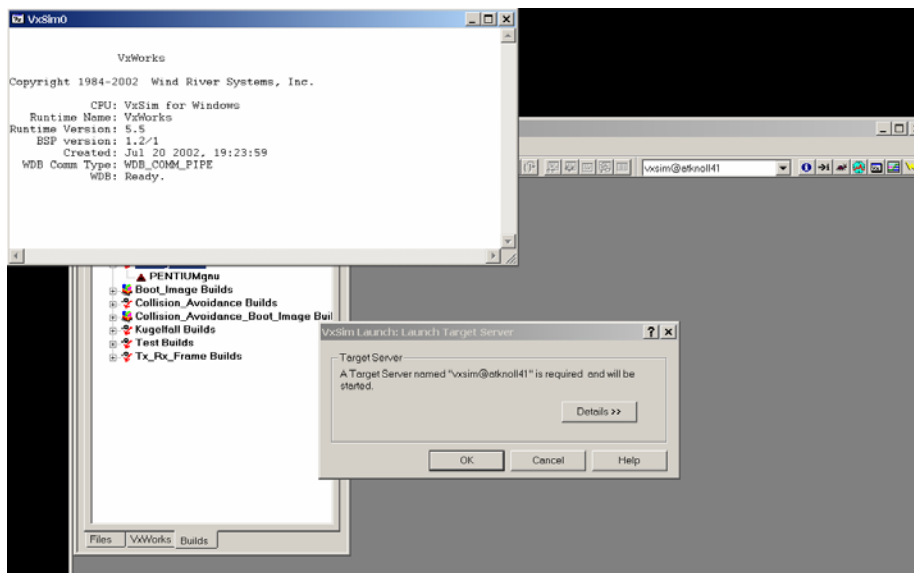


Abb.8 zweiter Dialog beim Starten des Simulators

In Abb.8 ist links oben bereits das Simulator-Fenster zu sehen. Ein gestarteter Simulator kommuniziert mit dem Host ebenfalls über einen Target-Server. Zu erkennen ist die Host-Simulator-Verbindung an einer kleinen roten Zielscheibe rechts unten in der Windows-Symbolleiste. Um eine ordnungsgemäße Funktion des Simulators zu gewährleisten, darf der Simulator nur einmal auf dem Hostrechner gestartet sein. Starten Sie nur den *integrierten Simulator*. Die Vollversion (*full simu-*

lator) ist nicht funktionsfähig. Der Simulator besitzt im Gegensatz zu Zielrechner keine Target-Shell. Einige *printf*-Ausgaben werden an das Simulator-Fenster gesendet; es sind aber keine Eingaben über das Simulator-Fenster möglich.

#### 2.1.4 Herunterladbare Anwendungen

Nach einem Kalt- oder Warmstart fordert jeder Zielrechner selbständig ein VxWorks-Image vom Rechner *atknoll41* an (gleich der Neuinstallation eines Betriebssystems). Nachdem das Image auf den Zielrechner übertragen wurde, kann eine (herunterladbare) Anwendung in das Image integriert werden (wie Installation eines Programms). Die Anwendung lässt sich nun durch Eingabe der Startroutine, meist *main*, in eine Shell oder über den Debugger starten. Der Startroutine können Parameter getrennt durch Kommata übergeben werden. Ein Aufruf mit Parametern wäre: *main par1, par2, ...*

#### Hinweise zum Verwalten eines Projekts

- **Dateien einem Projekt hinzufügen:**

Um eine neue Datei zu erstellen, klickt man auf *File->new*, wählt den Typ der Datei, den Namen des Projektes, zu dem sie gehören soll, und gibt einen Namen und den Speicherort für die Datei an. Eine bereits existierende Datei kann zu einem Projekt per Kontextmenü hinzugefügt werden. Dazu öffnet man per Rechtsklick auf das Arbeitsbereichfenster das Menü und wählt im sich öffnenden Menü den Punkt *add Files*. Dann kann die hinzuzufügende Datei ausgewählt werden.

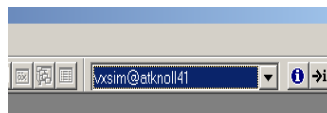
- **Das Build-Kommando ausführen**


Um eine Datei eines Projekts bzw. das ganze Projekt zu kompilieren, kann im Menü *Build* der Menüpunkt *Build* bzw. *Rebuild All* ausgewählt werden. Alternativ kann im Workspace-Fenster nach Rechtsklick auf das Projekt der Eintrag *Build All* bzw. *ReBuild All* ausgewählt werden.

Für die Übersetzung sind verschiedene Optionen nach Auswahl der Registerkarte *Builds* (Registerfeld unten rechts im Workspace-Fenster) einstellbar. Über die optionalen Einstellungen kann ein und dieselbe Anwendung für verschiedene Zielrechner kompiliert werden. Beispielsweise kann ein Projekt, das letztendlich auf einem Pentium-Target laufen soll, zu Testzwecken für den Simulator kompiliert werden. Dazu wählt man im Registerfeld *Builds* nach Rechtsklick auf das entsprechende Projekt den Eintrag *new Build* aus und selektiert beim Label *Default Build Spec for to* den Simulator (SIMNTgnu). Anschließend muss das Projekt noch neu kompiliert werden.

- **Objektmodule auf einen Zielrechner herunterladen**

Um eine kompilierte Anwendung auf einen Zielrechner zu laden, muss der entsprechende Rechner im Kombinationsfeld der IDE ausgewählt sein.




Erkennbar ist eine solche Auswahl an einem blauen Hintergrund im Kombinationsfeld der Tornado-IDE. Eine Möglichkeit zum Herunterladen der Anwendung besteht darin, nach Rechtsklick auf die Anwendung den Menüpunkt *download ‚ProjektName.out‘* auszuwählen. Eine zweite Möglichkeit wäre, auf das Symbol  in der Werkzeugleiste der IDE zu klicken. Im sich öffnenden Dialogfenster muss nun der Name der zu übertragenden Datei *ProjektName.out* gesucht werden. Nach Klicken auf Download startet der Download-Vorgang. In beiden Fällen wird nach erfolgreicher Übertragung keine Rückmeldung ausgegeben. Eine herunter geladene Anwendung lässt sich mit Rechtsklick auf das Projekt und anschließender Auswahl des Menü-

punktes *unload* ‚ProjektName.out‘ wieder vom Target entfernen. Beachten Sie, dass ein Projekt, das mit einer Toolchain für den Simulator kompiliert wurde, nicht auf einem Pentium-Target ausgeführt werden kann und umgedreht. Sollten Sie mehrere Projekte gleichzeitig auf ein Target laden, können Namenskonflikte bei gleichlautenden Routinen und Variablen auftreten.

**Beachten Sie:** erst beim Übertragen auf den Zielrechner werden die einzelnen Objektdateien endgültig verlinkt und entsprechende Fehlermeldungen bei fehlenden Funktionen erzeugt.

## 2.2 Shells

Wenn im Kombinationsfeld der Tornado-IDE ein Zielrechner ausgewählt wurde, kann zu diesem Rechner eine Host-Shell (WindShell) durch Klicken auf das Symbol  geöffnet werden. Es existieren neben Host-Shells auch Target-Shells. Eine Target-Shell wird automatisch beim der Verbindung zu einem Zielrechner geöffnet. Man kann mehrere Host-Shells in Verwendung haben, aber nur eine Target-Shell kann mit einem Zielrechner verbunden sein. Prinzipiell wird empfohlen, jegliche Shell-Kommunikation über Host-Shell abzuwickeln, weil deren Funktionsumfang größer ist und der Zielrechner durch die Verwendung von Host-Shells weniger beansprucht wird.

Allerdings besitzen Host-Shells gegenüber der Target-Shell einen großen Nachteil. Die Ausgabe von printf-Anweisungen bei Programmen mit nebenläufigen Prozessen erfolgt beim Start auf der Host-Shell nur teilweise und häufig nicht in der richtigen Reihenfolge. Ist die richtige Reihenfolge nötig, so ist ein Start aus der Targetshell nötig.

Shells bieten Zugang zu vielen Funktionalitäten des Betriebssystems. So können über sie global deklarierte C-Routinen aufgerufen werden sowie Variablen deklariert und initialisiert werden. Shells können zum Debugging einsetzen werden und natürlich ist der gesamte Funktionsumfang von VxWorks über Shells aufrufbar. Über Shells lassen sich neue Prozesse starten (spawn) und es kann ein Neustart des Rechners (reboot) veranlasst werden. Durch einen Neustart wird ein reines VxWorks-Image ohne Anwendungsdaten auf den Zielrechner übertragen, analog zu einer Neuinstallation des Betriebssystems.

VxWorks unterstützt auch eine Interpretierung von C-Code. So ist eine Eingabe von den meisten C-Ausdrücken direkt in der Eingabezeile möglich. Eine Shell liest dazu den Eingabestrom zeilenweise, parst die Zeilen, wertet sie aus und schickt ein Ergebnis an einen Ausgabestrom. Bis auf wenige Abweichungen wird dieselbe Syntax, wie sie von einem C-Compiler her bekannt ist, akzeptiert. Das System verarbeitet die Anweisungen exakt zu dem Ergebnis, dass während der Ausführung einer Anwendung erhalten werden würde. Die Möglichkeit, Anweisungen interpretativ auszuwerten, erspart das Kompilieren und Herunterladen. Ein Beispiel für den Routinen-Aufruf über Shells, ist der Aufruf der global deklarierten Hauptroutine Ihrer Anwendung.

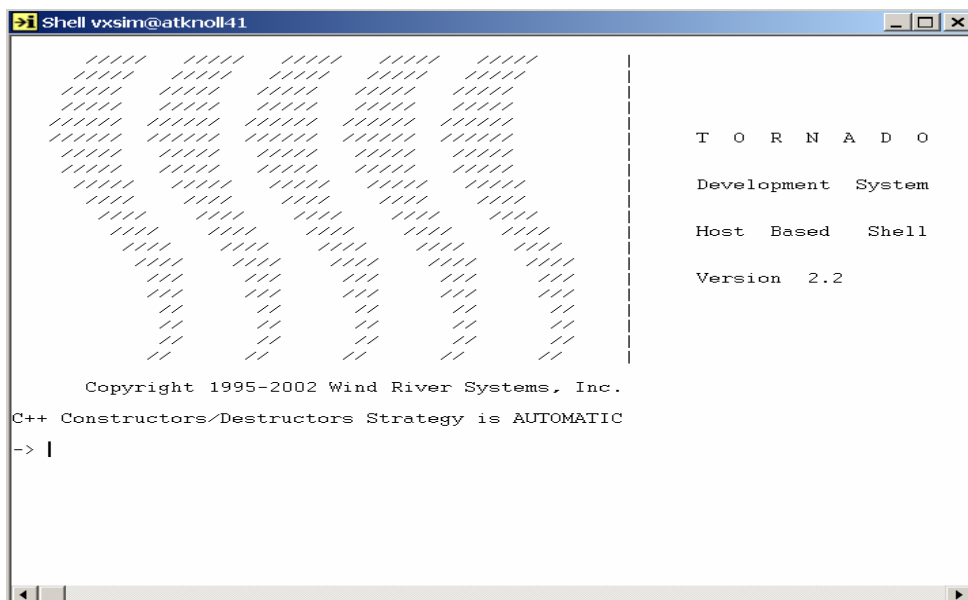

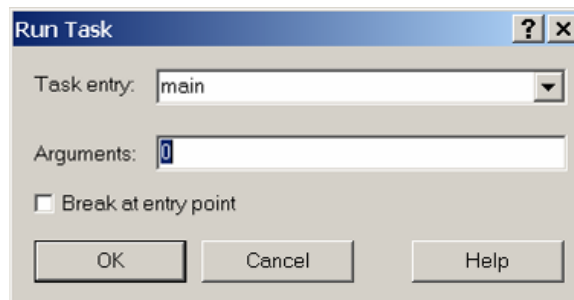


Abb.9 Host-Shell für die Kommunikation mit einem Target

## 2.3 Der Debugger

Nachdem der Debugger durch Klick auf das Symbol  gestartet wurde, sind weitere Schritte zum Starten der Anwendung durchzuführen. Aus dem Menü *Debug* muss der Menüpunkt *Run* ausgewählt werden. In das sich öffnende Fenster *Run Task*, das in Abbildung 10 zu sehen ist, müssen die Startroutine der Anwendung sowie eventuelle Übergabeparameter an die Anwendung eingegeben werden.



**Abb.10** Angabe der Startroutine und der Übergabeparameter beim Debugging

Soll ein anderer Prozess als der mit der Eintrittsroutine verbundene getestet werden, empfiehlt es sich, die Checkbox *Break at entry point* auszuwählen. Der Debugger wird daraufhin gleich nach dem Eintritt in die Startroutine angehalten. Nun kann mit der Taste F10 zeilenweise durch die Eintrittsroutine gesprungen werden bis alle Prozesse der Anwendung initialisiert sind. Anschließend kann über den Menüpunkt *Debug->Attach* der zu testende Prozess ausgewählt werden. Nach Klicken auf *Attach* springt der Debugger zum gewählten Prozess und bleibt dort in der ersten Zeile stehen. Erst jetzt werden Haltepunkte im gewählten Prozess berücksichtigt. Mit F5 kann der gestoppte Prozess fortgesetzt werden. Beachten Sie, dass nur der momentan an den Debugger „angebundene“ Prozess mit seinen Haltepunkten überwacht wird. Alle anderen Prozesse, zu denen bei VxWorks letztlich auch Unterbrechungsroutrinen (interrupts) gehören, laufen ohne Störungen weiter.

Haltestellen innerhalb von Unterbrechungsroutrinen können nur im Systemmodus des Debuggers berücksichtigt werden, der leider für das Praktikum nicht zur Verfügung steht. Sollen Unterbrechungsroutrinen getestet werden, ist man gezwungen, sich mit anderen Hilfsmitteln (z.B. globale Variablen) zu verwenden.

Da das Setzen von *printf*-Anweisungen insbesondere in Unterbrechungsroutrinen eigene Tücken mit sich bringt, ist die Verwendung von *printf*-Anweisungen zur Überwachung nicht zu empfehlen. Zum Ersten werden *printf*'s stets ungepuffert ausgeführt. Die Folge ist, dass die Abarbeitung einer *printf*-Anweisung auf Interruptniveau von statten geht. Die Unterbrechungsroutrine wird dadurch nicht unerheblich verlängert, was wiederum der Philosophie einer Unterbrechung widerspricht und sich besonders bei hohen Aufrufzahlen negativ auswirkt. Zum Zweiten kann man allgemein beim Markieren von Programmstellen mit *printf*'s nicht sicher sein, dass ein Fehler wirklich vor einer nicht mehr ausgegebenen *printf*-Anweisung auftrat. Der Grund ist die Multiprozess-Umgebung, in der eine VxWorks-Anwendung läuft. Wird beispielsweise das *printf* in einem Prozess noch berücksichtigt und ein anderer Prozess kommt aufgrund höherer Priorität gleich nach der Berücksichtigung zum Zuge und verursacht einen Systemabsturz, dann wird man die ausgeführte *printf*-Anweisung nicht zu sehen bekommen. Die Fehlersuche wird deshalb höchstwahrscheinlich an der falschen Stelle beginnen.

Um Variablen oder Prozesse zur Überwachung hinzuzufügen, öffnet man das Kontextmenü durch einen Rechtsklick auf die entsprechende Variable im Programmcode und wählt *add to watch* aus. Die Variable wird in das Beobachtungsfenster aufgenommen. Für lokale Variable gibt es eine zweite Beobachtungsmöglichkeit. Das Variablen-Fenster zeigt alle lokalen Variablen der aktuellen Routine an. Beide Fenster erreichen Sie über den Menüpunkt *Debug->Debug Windows*. Um Haltepunkte zu setzen, öffnet man an der jeweiligen Stelle im Code das Kontextmenü (Rechtsklick) und wählt *Breakpoint* aus. Eine Anwendung lässt sich auch im Einzelschritt-Modus (single step) ausführen, z.B. mit der Taste F10 (ohne Eintauchen in Routinen) oder F11 (mit Eintauchen in Routinen). Der Debugger ermöglicht noch eine Menge weitere Funktionen, die im *Tornado User's Guide* Kapitel 10 dokumentiert sind.



Sollten Sie den Quellcode Ihrer Anwendung mit der Taste F11 testen, dann könnte sich bei der Ausführung von bestimmten Zeilen ein Fenster mit Assembler-Anweisungen öffnen. Bei den betreffenden Zeilen handelt es sich um Anweisungen, deren Debug-Information, die für den Zugang zum Quellcode immer notwendig ist, im VxWorks-Image liegt. Grundsätzlich kann das VxWorks-Image als fehlerfrei betrachtet werden und die entsprechenden Zeilen können mit F10 übersprungen werden. Möchten Sie dennoch einen Blick „hinter“ den Code der betreffenden Zeilen werfen, dann müssen Sie nach dem Starten des Debuggers folgende Einstellung vornehmen:

- Menüpunkt *Debug->Source Search Path* auswählen,
- Eintragen des Pfades `\\Atnoll41\Boot Image\default` in das Fenster *Directories*.

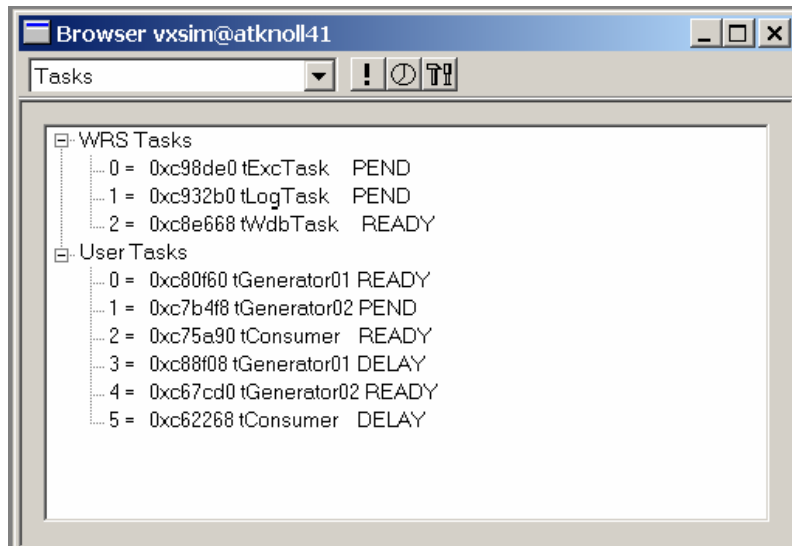
## 2.4 Der Browser

Mit Hilfe des Browsers kann man den Zustand eines Zielrechners überwachen. Mit dem Browser sind Informationen, die über Kommandozeile in Textform abrufbar sind, graphisch darstellbar. Im Hauptfenster werden z.B. aktive Prozesse oder der Speicherverbrauch eines Zielrechners ausgegeben. Insgesamt können im Browser-Fenster Informationen zu

- Prozess-Prioritäten und Registernutzung,
- Semaphoren,
- Nachrichtenwarteschlangen (Message-Queues),
- Watchdog-Timern,
- Nutzung der Prozess-Stacks,
- Nutzung der CPU durch die Anwendungsprozesse,
- Speichernutzung auf dem Zielrechner,
- Aufbau und Symbolen von allen Objektmodulen, aus denen die Anwendung auf dem Zielrechner zusammengesetzt ist,

angezeigt werden. Der Browser lässt sich, nachdem ein Zielrechner ausgewählt wurde, mit Klick auf  , oder durch die Menüauswahl *Tools->Browser* starten. Doch Vorsicht: Alle angezeigten Informationen sind Schnappschüsse (statisch). Sie müssen interaktiv durch Klick auf  oder periodisch (automatisch) aktualisiert werden. Es kann also sein, dass auf dem Zielrechner kurzzeitig vorhandene Informationen nicht rechtzeitig zur Anzeige kommen, weil sie zum Zeitpunkt der Auffrischung des Browser-Fensters nicht mehr vorhanden sind. Detaillierte Hinweise zur Benutzung des Browsers entnehmen Sie bitte den Online Manuals: *Tornado User's Guide, Kapitel 9*.


Um dynamische Abläufe eines Zielrechners zu erfassen, wird WindView eingesetzt, dessen Möglichkeiten im nächsten Abschnitt angesprochen werden. Abbildung 11 zeigt das Browser-Fenster, das sich nach Aktivierung des Browsers öffnet.

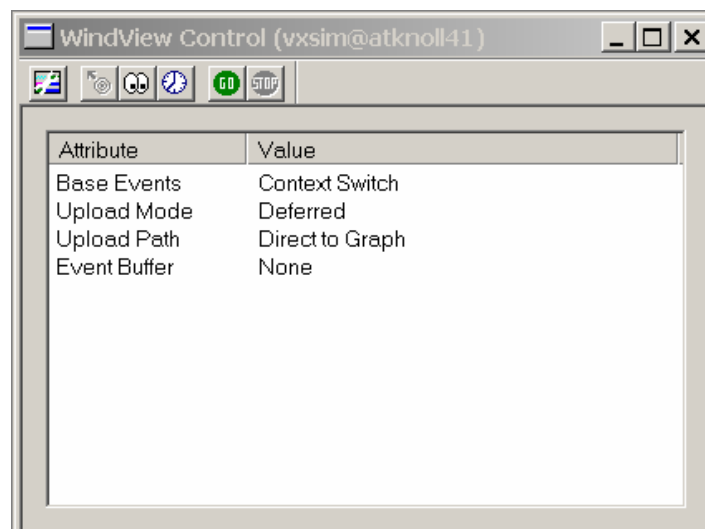


**Abb.11** Das Browser-Fenster nach Aktivierung des Browsers







## 2.5 WindView

WindView ist ein Logikanalysator für VxWorks, mit dem dynamische Abläufe Ihrer Anwendung untersucht werden können. Durch WindView können beispielsweise Laufzeiten von Prozessen, Verklemmungen (Deadlock) und CPU-Verhungern sichtbar gemacht werden. Das Zeitdiagramm im WindView-Graphen-Fenster (vgl. Abb. 13) kann auf die speziellen Erfordernisse der Analyse zugeschnitten werden. So ist es möglich, nur Prozesse und Ereignisse zu berücksichtigen, die im Moment von Interesse sind. An allen Host-Rechnern steht die Vollversion von WindView zur Verfügung. Das bedeutet, WindView ist auch für Hardware-Zielrechner einsetzbar. (Die integrierte Version könnte nur in Verbindung mit dem Simulator verwendet werden.)

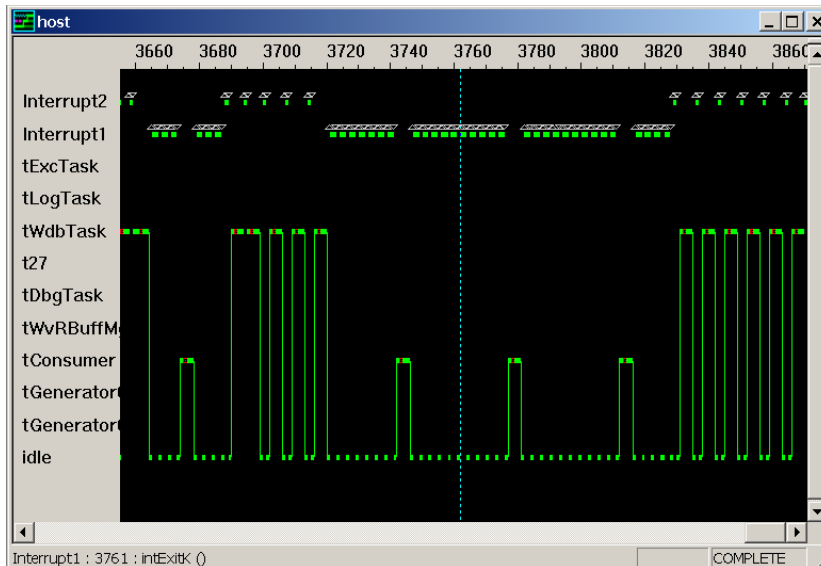
WindView wird über das Symbol  oder den Menüpunkt *Tools->WindView->Launch* gestartet. Es öffnet sich ein Fenster wie in Abbildung 12.



**Abb.12** Das WindView-Kontroll-Fenster

Mit Klick auf  leitet man eine Datensammlung auf dem Zielrechner ein. Ein paar Sekunden warten, anschließend auf den Update-Button  klicken und der Status der Datensammlung wird aktualisiert. Wieder kurz warten, auf Stop  klicken und anschließend den Upload-Button  benutzen, um die Daten auf den Host zu laden. Mit den Zoom-Buttons   lässt sich der Auszug in der Umgebung der gestrichelten, blauen Linie vergrößern oder verkleinern.

Zum Schnelleinstieg in WindView lesen Sie bitte im *Tornado Getting Started Guide, Kapitel 3* nach. Für detaillierte Informationen ist der *WindView User's Guide* zu empfehlen.



**Abb.13** Diagramm im WindView-Graphen-Fenster

Für den Simulator wurde keine Funktionalität für Zeitstempel ins entsprechende VxWorks-Image eingebunden. Deshalb erscheint nach dem Laden der im Zielrechner gesammelten Daten im oberen Teil des Graphen-Fensters eine Ereignisskala. Jedes Ereignis hat den gleichen Abstand zum vorherigen, weshalb keine Rückschlüsse auf Laufzeiten von Prozessen bzw. Unterbrechungsroutinen möglich sind.


Für Zielrechner ist die Zeitstempelfunktionalität eingebunden. Im oberen Teil des Graphen-Fensters erscheint nach dem Laden eine Zeitskala mit einer Auflösung bis in den Millisekundenbereich. Durch die Zeitskala und mit Hilfe von senkrechten, braunen Linien, die mit dem Mauszeiger über das Diagramm bewegt werden können, sind sehr genaue Zeitmessungen möglich. Bei solchen Zeitmessungen ergibt sich eine Zeitdifferenz aus der Position der ersten und der zweiten braunen Linie. Die Zeitdifferenz wird unten links im WindView-Graphen-Fenster angezeigt.

## 2.6 Die zentrale Tornado-Verwaltung WindView

Spätestens ab Aufgabe 4 muss eine Verbindung zu einem der Zielrechner hergestellt werden. Führen Sie dazu im Vorfeld die unter 2.6.2. aufgelisteten Schritte durch.

### 2.6.1 Nutzung der zentralen Tornado-Verwaltung

Um auszuschließen, dass auf mehreren Entwicklungsrechnern gleichzeitig eine Verbindung zu dem selber Zielrechner aufgebaut wird, wird zur Verwaltung der Verbindungen zwischen Entwicklungs- und Zielrechnern eine zentrale Tornado-Verwaltung genutzt. Diese Verwaltung befindet sich auf dem zentralen Praktikumsrechner *atknoll41*. Eine gültige Host-Target-Verbindung wird nur aufgebaut, wenn eine Verbindung zu demselben Zielrechner auf dem Rechner *atknoll41* noch nicht registriert ist. Erst nachdem ein Entwicklungsrechner seine Verbindung mit dem Zielrechner abgebaut hat, kann ein anderer Entwicklungsrechner eine Verbindung mit dem Zielrechner aufbauen.

Sie erkennen eine aufgebaute Verbindung am Kombinationsfeld in der Symbolleiste der Tornado-IDE. Im Kombinationsfeld erscheint eine solche Host-Target-Verbindung mit dem Namen des Entwicklungsrechners gefolgt von einem @ und dem Zielrechnernamen, z.B. *atknoll67@Kugelfall* oder *atknoll68@Aufzug2*. Falls ein anderer Praktikumssteilnehmer versucht, eine Verbindung zu einem belegten Zielrechner aufzubauen, wird in der sich öffnenden Target-Shell kein Prompt erscheinen. Der Versuch eine Verbindung aufzubauen ist damit misslungen und muss ordnungsgemäß abgeschlossen werden. Zum ordnungsgemäßen Abschluss ist der betreffende Target-Server über das Symbol  in der unteren Bildschirmleiste rechts auszuwählen und nach Rechtsklick auf das Symbol zu schließen. Es ist leider möglich, von jedem Entwicklungsrechner aus eine bestehende Verbindung aus dem Kombinationsfeld der Symbolleiste auszuwählen und hierfür eine Host-Shell zu öffnen. Hierüber können dann Kommandos zum Zielrechner einer anderen Gruppe abgesetzt werden, was eher unangenehm ist und deshalb unterlassen werden sollte. Benutzen sie nur Verbindungen zu von ihnen genutzten Zielrechnern und geben sie diese sobald sie die Rechner nicht mehr benutzen auch wieder frei.