INSTYTUT ŁĄCZNOŚCI
PAŃSTWOWY INSTYTUT BADAWCZY

Zakład Systemów Zasilania (Z-5)

Sprawozdanie nr 306/Z5

Rdzeń modułowego system czasu rzeczywistego do profesjonalnych aplikacji hybrydowych systemów zasilania i systemów automatycznego nadzoru.

Praca nr 05300017

Warszawa grudzień 2007

Praca nr 05300017

Słowa kluczowe (maksimum 5 słów): oprogramowanie, systemy czasu rzeczywistego, systemy nadzoru

Kierownik pracy: mgr inż. Maciej Kozyra

Wykonawcy pracy:

mgr inż.	Maciej	Kozyra	Z5
dr inż.	Robert	Samborski	Z5
mgr inż.	Edward	Chrustowski	Z5
inż.	Paweł	Kliś	Z5
technik	Andrzej	Stułka	Z5
technik	Krzysztof	Kanicki	Z5
technik	Genowefa	Dziuba	Z5

Kierownik Zakładu: inż. Paweł Kliś

Spis treści

Wprowadzenie

W pracy wykonano środowisko do programowania mikrokontrolerów wykorzystywanych w układach zasilania i nadzoru układów zasilania w oparciu o narzędzia o otwartym kodzie i publicznie dostępne bez konieczności uiszczania opłat za ich używanie oraz przy jego użyciu skompilowano i uruchomiono kilka systemów czasu rzeczywistego przeznaczonego dla systemów typu embeded.

W sprawozdaniu opisano po krótko analizowane systemy i zestawiono ich właściwości i platformy sprzętowe i procesorowe w których mogą zostać zaimplementowane. Oprócz tego dokonano analizy i przystosowano do użytku dla potrzeb systemów zasilania system FreeRTOS (zaaportowano) na procesory AT91SAM7S firmy atmel, SRT912 firmy SG i LPC2138 firmy Philips.

Pojęcie systemu czasu rzeczywistego

„System czasu rzeczywistego to taki, w którym wynik przetwarzania nie zależy tylko i wyłącznie od jego logicznej poprawności, ale również od czasu, w jakim został osiągnięty. Jeśli nie są spełnione ograniczenia czasowe, mówi się, że nastąpił błąd systemu.” (patrz np. [1]).
W literaturze można spotkać również inne definicje, które w mniej lub bardziej obrazowy sposób oddają istotę tego typu systemów. Niektóre z nich brzmią następująco:

- Tryb przetwarzania w czasie rzeczywistym jest takim trybem, w którym programy przetwarzające dane napływające z zewnątrz są zawsze gotowe, a wynik ich działania jest dostępny nie później niż po zadanym czasie. Moment nadejścia kolejnych danych może być losowy (asynchroniczny) lub ściśle określony (synchroniczny) [2].
- System czasu rzeczywistego jest systemem interaktywnym, który utrzymuje ciągły związek z asynchronicznym środowiskiem, np. środowiskiem, które zmienia się bez względu na system, w sposób niezależny [2].
- Oprogramowanie czasu rzeczywistego odnosi się do systemu lub trybu działania, w którym przetwarzanie jest przeprowadzane na bieżąco, w czasie wystąpienia zewnętrznego zdarzenia, w celu użycia rezultatów przetwarzania do kontrolowania lub monitorowania zewnętrznego procesu [2].
- System czasu rzeczywistego odpowiada w sposób przewidywalny (w określonym czasie) na bodźce zewnętrzne napływające w sposób nieprzewidywalny [2].System mikrokomputerowy działa w czasie rzeczywistym, jeżeli wypracowane przez ten system decyzje są realizowane w tempie obsługiwanego procesu. Inaczej mówiąc, system działa w czasie rzeczywistym, jeżeli czas reakcji systemu jest niezauważalny przez proces (decyzja jest wypracowana we właściwym czasie) [3].

Powyższe definicje pozwalają wysnuć wniosek, że samo określenie „czasu rzeczywistego” jest pojęciem w pewnym sensie względnym. Niech będą dane dwa systemy działające w czasie rzeczywistym:

- odtwarzacz mp3;
- system sterowania przetwarzaniem energii;

W pierwszym przypadku, strumienie obrazu i dźwięku są dekodowane przez odtwarzacz jako wynik odpowiedzi na rozkazy wydane przez użytkownika. Jednakże rozkazy mogą zostać tak gwałtownie wydawane, że dekoder nie będzie w stanie ich natychmiast obsłużyć. Przekroczy swoje ograniczenia czasowe. Jako rezultat lub kara za takie postępowanie, pojawią się chwilowe, lecz niestety słyszalne, zniekształcenia dźwięku. Odtwarzacz mp3 oczywiście nie przestanie funkcjonować. Dalej będzie poprawnie spełniał swoje zadania.

W przypadku systemów zasilania bardzo istotny jest tzw. czas reakcji systemu, czyli przedział czasu potrzebny systemowi na wypracowanie decyzji (sygnału wyjściowego) w odpowiedzi na zewnętrzny bodziec (sygnał wejściowy). System umożliwiający sterowanie przetworzeniem energii w czasie rzeczywistym, aby mógł poprawnie spełniać swoje zadania, musi bezwarunkowo spełnić bardzo rygorystyczne ograniczenia czasowe. Jeśli np. nowe parametry z jakiś powodów nie mogą zostać wyliczone odpowiednio szybko, system stara się jak najlepiej uśrednić swoje obliczenia i wyprowadzić ich wyniki na zewnątrz.

Można dojść do wniosku, że zależnie od roli i przeznaczenia danej grupy aplikacji czasu rzeczywistego, ograniczenia czasowe są koniecznością, której niespełnienie prowadzi w najgorszym przypadku do nieodwracalnych i tragicznych skutków oraz tych, w których czas wykonania nie jest tak krytyczny i dopuszcza się pewne odstępstwa. Najczęściej systemy czasu rzeczywistego dzieli się na dwie grupy:

Rygorystyczne (twarde, ang. Hard real – time systems) - gwarantują terminowe wypełnianie krytycznych zadań. Osiągnięcie tego celu wymaga ograniczenia wszystkich opóźnień w systemie, poczynając od odzyskiwania przechowywanych danych, a kończąc na czasie zużywanym przez system na wypełnienie dowolnego zamówienia. Takie ograniczenia czasu wpływają na dobór środków, w które są wyposażane rygorystyczne systemy czasu rzeczywistego. Wszelkiego rodzaju pamięć pomocnicza jest na ogół bardzo mała albo nie występuje wcale. Wszystkie dane są przechowywane w pamięci o krótkim czasie dostępu lub w pamięci, z której można je tylko pobierać (ang. read-only memory- ROM). Pamięć ROM jest nieulotna, tzn. zachowuje zawartość również po wyłączeniu dopływu prądu elektrycznego; większość innych rodzajów pamięci jest nietrwała. Systemy te nie mają również większości cech nowoczesnych systemów operacyjnych, które oddalają użytkownika od sprzętu, zwiększając niepewność odnośnie do ilości czasu zużywanego przez operacje. Na przykład prawie nie spotyka się w systemach czasu rzeczywistego pamięci wirtualnej. Dlatego rygorystyczne systemy czasu rzeczywistego pozostają w konflikcie z działaniem systemów z podziałem czasu i nie wolno ich ze sobą mieszać. Przykładem może być system nadzoru układu zasilania.

Łagodne (miękkie, ang. Soft real – time systems) – są mniej wymagające. W nich krytyczne zadanie do obsługi w czasie rzeczywistym otrzymuje pierwszeństwo przed innymi zadaniami i zachowuje je aż do swojego zakończenia. Podobnie jak w rygorystycznym systemie czasu rzeczywistego opóźnienia muszą być ograniczone - zadanie czasu rzeczywistego nie może w nieskończoność czekać na usługi jądra. Łagodne traktowanie wymagań dotyczących czasu rzeczywistego umożliwia godzenie ich z systemami innych rodzajów. Jednak użyteczność łagodnych systemów czasu rzeczywistego jest bardziej ograniczona niż systemów rygorystycznych. Ponieważ nie zapewniają one nieprzekraczalnych terminów, zastosowanie ich w przemyśle i robotyce jest ryzykowne. Niemniej jednak istnieje kilka dziedzin, w których są one przydatne. Są to np. techniki multimedialne, kreowanie sztucznej rzeczywistości, zaawansowane projekty badawcze w rodzaju eksploracji podmorskich lub wypraw planetarnych. Znajdują one swoje miejsce wszędzie tam, gdzie istnieje potrzeba systemów o bardziej rozbudowanych możliwościach .

Oczywiście w literaturze można spotkać również pewną pośrednią grupę, tzw. firm real – time systems (patrz np. [4]). Tutaj nie ma żadnej korzyści jeśli nastąpi spóźnienie w dostarczaniu usług. Nie ma też żadnej groźby związanej z takim przypadkiem.

Definicje

System czasu rzeczywistego

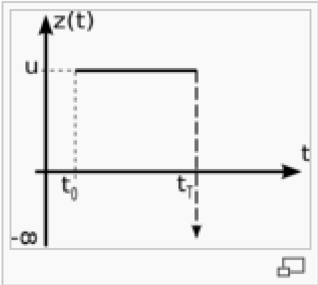
Definicja systemu czasu rzeczywistego została zaczerpnięta z wikipedii. System czasu rzeczywistego (ang. real-time system), to urządzenie techniczne, którego wynik i efekt działania jest zależny od chwili wypracowania tego wyniku. Istnieje wiele różnych definicji naukowych takiego systemu. Ich wspólną cechą jest zwrócenie uwagi na równoległość w czasie zmian w środowisku oraz obliczeń realizowanych na podstawie stanu środowiska. Z tego wyścigu dwóch stanów: zewnętrznego i wewnętrznego, wynikają kryteria ograniczające czas wypracowywania wyniku.

Funkcja zysku

Dla teorii i praktyki systemów czasu rzeczywistego przydatne jest pojęcie funkcji zysku. Funkcja zysku jest funkcją zależną przede wszystkim od czasu i określa korzyść ze zrealizowania zadania przez system. Korzyść niekoniecznie jest wielkością wymiarowaną. Źródłem ograniczeń czasowych są zazwyczaj zjawiska fizyczne zachodzące w świecie rzeczywistym. Zadanie zostało przez system zrealizowane poprawnie, jeśli z chwilą zakończenia tego zadania wartość funkcji zysku jest większa od zera.

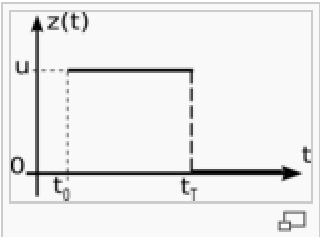
Funkcję zysku można określać nie tylko dla systemów czasu rzeczywistego. Następująca funkcja zysku: $y = a$ jest typowym przykładem dla systemu innego niż czasu rzeczywistego - zysk jest zawsze taki sam, niezależnie od momentu uzyskania wyniku.

W zdecydowanej większości praktycznie rozpatrywanych przypadków funkcja zysku jest nierosnąca - korzyść z wykonania zadania nie rośnie z upływem czasu. Szczególnym przypadkiem gdy nie jest to spełnione, mogą być na przykład operacje giełdowe. Dla systemów czasu rzeczywistego charakterystyczne są trzy funkcje patrz rysunki poniżej:



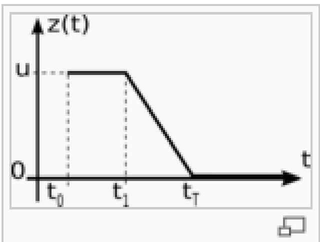
$$z(t) = \begin{cases} u & t_0 < t < t_T \\ -\infty & t \geq t_T \end{cases}$$

Rysunek 1: Hard



$$z(t) = \begin{cases} u & t_0 < t < t_T \\ 0 & t \geq t_T \end{cases}$$

Rysunek 2: Firm



$$z(t) = \begin{cases} u & t_0 < t < t_1 \\ \frac{dz(t)}{dt} = const \wedge \frac{dz(t)}{dt} < 0 & t_1 < t < t_T \\ 0 & t \geq t_T \end{cases}$$

Rysunek 3: Soft

Z tych trzech funkcji wyprowadza się przedstawiony dalej podział systemów czasu rzeczywistego na "hard"(1.), "firm"(2.) i "soft"(3.).

t_0 - to chwila zlecenia zadania systemowi, uznawana za początek realizacji tego zadania,

t_T to najpóźniejsza chwila w której przetwarzanie może zostać zakończone (ang. deadline),w przypadku (3.) funkcja z(t) jest ciągła w punktach t_1 i t_T,

Dla $t <= t_0$ wartości funkcji zysku nie określa się, gdyż oczywiście nie jest to sytuacja realizowana fizycznie.

Praktyczne uproszczenie systemu czasu rzeczywistego

W systemie czasu rzeczywistego przekształcanie danych przesyłanych do lub z zewnętrznego środowiska zachodzi w deterministycznie określonym czasie. Stosuje się pojęcie terminu (ang. deadline), oznaczające najdłuższy dopuszczalny czas reakcji systemu na wystąpienie zdarzenia. System czasu rzeczywistego nie musi być szybki - istotne jest jedynie, aby jego działania spełniały narzucone ograniczenia czasowe.

System informatyczny czasu rzeczywistego

Często pod pojęciem "system czasu rzeczywistego" rozumie się systemy zbudowane z wykorzystaniem komputera, pracującego pod kontrolą systemu operacyjnego czasu rzeczywistego. W skład takiego systemu włącza się także jego niezbędne otoczenie, takie jak deterministyczne sieci transmisyjne (np. Fip, Modbus, Genius, CAN i in.), układy wejściowe i wyjściowe oraz urządzenia kontrolowane przez komputer (np. roboty).

Aby system składający się z komponentów był systemem czasu rzeczywistego, konieczne jest spełnianie wymogów systemu czasu rzeczywistego przez każdy z komponentów. W przypadku systemów informatycznych oznacza to, że zarówno sprzęt, system operacyjny, jak i oprogramowanie aplikacyjne muszą gwarantować dotrzymanie zdefiniowanych ograniczeń czasowych.

W realizacji oprogramowania działającego w czasie rzeczywistym niezbędna jest analiza wydajności działania aplikacji.

Podział systemów czasu rzeczywistego

Praktyczny podział systemów czasu rzeczywistego wynika z opisanych wcześniej trzech charakterystycznych teoretycznych funkcji zysku.

- systemy o ostrych ograniczeniach czasowych (ang. hard real-time) - gdy przekroczenie terminu powoduje poważne, a nawet katastrofalne skutki, jak np. zagrożenie życia lub zdrowia ludzi, uszkodzenie lub zniszczenie urządzeń, przy czym nie jest istotna wielkość przekroczenia terminu a jedynie sam fakt jego przekroczenia,
- systemy o mocnych ograniczeniach czasowych (ang. firm real-time) - gdy fakt przekroczenia terminu powoduje całkowitą nieprzydatność wypracowanego przez system wyniku, jednakże nie oznacza to zagrożenia dla ludzi lub sprzętu; pojęcie to stosowane jest głównie w opisie teoretycznym baz danych czasu rzeczywistego,
- systemy o miękkich lub łagodnych ograniczeniach czasowych (ang. soft real-time) - gdy przekroczenie pewnego czasu powoduje negatywne skutki tym poważniejsze, im bardziej ten czas został przekroczony; w tym przypadku przez "negatywne skutki" rozumie się spadek funkcji zysku aż do osiągnięcia wartości zero w chwili t_T.

Niektórzy autorzy negują powyższy podział, uznając za systemy czasu rzeczywistego wyłącznie te o ostrych ograniczeniach czasowych. Część autorów za systemy czasu rzeczywistego uznaje tylko te, które są weryfikowalne metodami formalnymi lub zawężając jeszcze bardziej: tylko te, które już zostały zweryfikowane pozytywnie. Dlatego w tym ujęciu powszechnie stosowane określenie, że dane zadanie jest wykonywane "w czasie rzeczywistym", jest traktowane jako nadużycie.

Narzędzia

Środowiska Programistyczne

W pracy przebadano funkcjonalność pięciu otwartych środowisk programistycznych umożliwiających tworzenie oprogramowania systemów czasu rzeczywistego ze szczególnym uwzględnieniem systemów głęboko osadzonych z procesorami ARM. Poniżej opisujemy pokródce te środowiska ich zawartość i platformy systemowe na których są posadzone.

WinARM

Winarm jest zbiorem narzędzi umożliwiających kompilowanie i programowanie mikrokontrolerów z procesorem ARM na platformie Microsoft Windows. W jego skład wchodzą:

- GNU-C/C++-kompliator (krosskompilator,linker asembler).Skrypty wpierające następujące konfiguracje procesora ARM-Mode, Thumb-Mode and Mixed(ARM/Thumb)-Mode, little/big-endian i floating point-emulation.
- GNU-Binutils 2.16
- newlib
- newlib-lpc
- GNU-utils
- pliki nagłówkowe ARM (definicje rejestrów)
- Przykładowe aplikacje
- Edytor „Programmers Notepad”
- lpc21isp narzedzie do programowania mikrokontrolerów LPC
- Bray Terminal
- Insight-GDB
- GDB
- OCDRemote (Wiggler-gdb interface)

GNU ARM toolchain

http://www.scienceprog.com/gnuarm-for-arm-microcontrollers/

Jest to środowisko crossplatformowe otwarte środowisko programistyczne umożliwiające tworzenie oprogramowania dla różnych procesorów ARM. Powyższy link opisuje instalacje i przykłady użycia środowiska dla systemu operacyjnego Windows.

CodeSourcery

http://www.codesourcery.com/gnu_toolchains/arm

CodeSourcery w porozumieniu za ARM Ltd. Wprowadził udoskonalenia do GNU Toolchain'a dla procesorów ARM i wydaje regularnie nowe wersje tego narzędzia.Sourcery G++ Lite Edition wspiera kompilacje ARM, Thumb, and Thumb-2 dla wszystkich architektur w użyciu wraz z 7-dmą wersją architektury procesora ARM.

YAGARTO

http://yagarto.de/

YAGARTO jest kolejną mutacją gnu ARM toolchain, jest podzielona na trzy pakiety:Open On-Chip Debugger / wsparcie dla J-Link/SAM-ICE GDB Serwer Binutils, Newlib, kompilator GCC i debugger Insight.

Biblioteki wykonawcze Platformy Eclipse, Eclipse CDT and CDT pluginy dla debudera GDB. W porównaniu do innych pakietów nie jest oparta na Cygwinie, pracuje z Eclipsem. Niestety nie działa z systemem windows98.Nie wspiera też procesorów z rdzeniem Cortex-M3.

GNU ARM Toolchain - Kompilacja ze źródeł pod Linuxem

Do skompilowania środowiska programistycznego umożliwiającego tworzenie oprogramowania dla systemów zasilania konieczne jest ściągniecie ze strony http://gnuarm.org/files.html następujących pakietów:

- binutils-2.17.tar.bz2 [13.1MB]
- gcc-4.2.0.tar.bz2 [42.0MB]
- newlib-1.15.0.tar.gz [10.2MB]
- insight-6.6.tar.bz2 [21.5MB]

W załączniku 1 załączono listę pakietów w jakie było wyposażone środowisko systemu umożliwiające właściwe skompilowanie narzędzi używanych w pracy.

Kompilację przeprowadzamy w następujący sposób:

1.cd [binutils-build]

2.[binutils-source]/configure --target=arm-elf --prefix=[toolchain-prefix] --enable-interwork --enable-multilib --with-float=soft

3.make all install

4.export PATH="$PATH:[toolchain-prefix]/bin"

5.cd [gcc-build]

6.[gcc-source]/configure --target=arm-elf --prefix=[toolchain-prefix] --enable-interwork --enable-multilib --with-float=soft --enable-languages="c,c++" --with-newlib --with-headers=[newlib-source]/newlib/libc/include

7.make all-gcc install-gcc

8.cd [newlib-build]

9.[newlib-source]/configure --target=arm-elf --prefix=[toolchain-prefix] --enable-interwork --enable-multilib --with-float=soft

10.make all install

11.cd [gcc-build]

12.make all install

13.cd [gdb-build]

14.[gdb-source]/configure --target=arm-elf --prefix=[toolchain-prefix] --enable-interwork --enable-multilib --with-float=soft

15.make all install

Po tych operacjach powinniśmy jeszcze dodać tak wykonane środowisko do naszych zmiennych systemowych.

Kompilator

GCC

GCC (ang. GNU Compiler Collection) to zestaw kompilatorów do różnych języków programowania rozwijany w ramach projektu GNU i udostępniany na licencji GPL oraz LGPL. GCC jest podstawowym kompilatorem w systemach uniksopodobnych przy czym szczególnie ważną rolę odgrywa w procesie budowy jądra Linuksa. Początkowo skrótowiec GCC oznaczał GNU C Compiler, ponieważ był to kompilator wyłącznie do języka C. Do kompilacji systemów czasu rzeczywistego wykorzystano wyłącznie kompilator C i C++.

IAR

IAR Systems przygotowało środowisko programistyczne dostosowane do tworzenia oprogramowania dla systemów wbudowanych na platformie Windows. W pracy nie skorzystano z niego z powodu ograniczenia na rozmiar kompilowanego kodu do 8kB. W praktyce testowane systemy zajmowały powyżej 32KB kodu po kompilacji. Drugim powodem dla którego nie zastosowano kompilatora IAR był brak wsparcia dla procesorów STR912 we wczesnej fazie prób z systemami czasu rzeczywistego.

KEIL

Firma Keil™, zajmuje się produkcją kompilatorów, makro asemblerów, kreneli czasu rzeczywistego, debugerów , symulatorów, zintegrowanych środowisk programistycznych płyt ewaluacyjnych i emulatorów dla firmy ARM® na procesory ARM7/ARM9/Cortex-M3. W pracy użyto wersji testowej środowiska RealView. Obecnie Keil stało się częścią ARM Ltd. I rozwija swoje produkty na jej potrzeby. Uważa się że jest to najlepsze komercyjne oprogramowanie przeznaczone na procesory ARM. Firma Keil na razie nie przewiduje wykonania tego środowiska na platformy Linux'owo Uniksowe.

Programator i jego oprogramowanie

Sam kompilator nie wystarcza do uruchomienia dowolnego oprogramowania nią mikrokontrolerze, konieczne jest urządzenie umożliwiające zaprogramowanie jego treści do jakiegoś nośnika. Nośnikiem tym może być ROM,EPROM,pamięć masowa, pamięć flash itp. W przypadku mikrokontrolerów wykorzystywanych w pracy jądro systemu czasu rzeczywistego kompilowane na stacji roboczej jest następnie zapisywane w pamięci flash mikrokontrolera. W przypadku platformy EP9302 mikroprocesor nie posiada wewnętrznej pamięci flash, ale pamięć zewnętrzną która jest programowana za pomocą innej metody niż opisane poniżej.

Programowanie mikrokontrolerów firmy arm firmy atmel może być wykonane z poziomu SO windows za pomocą oprogramowania Sam-Ba.

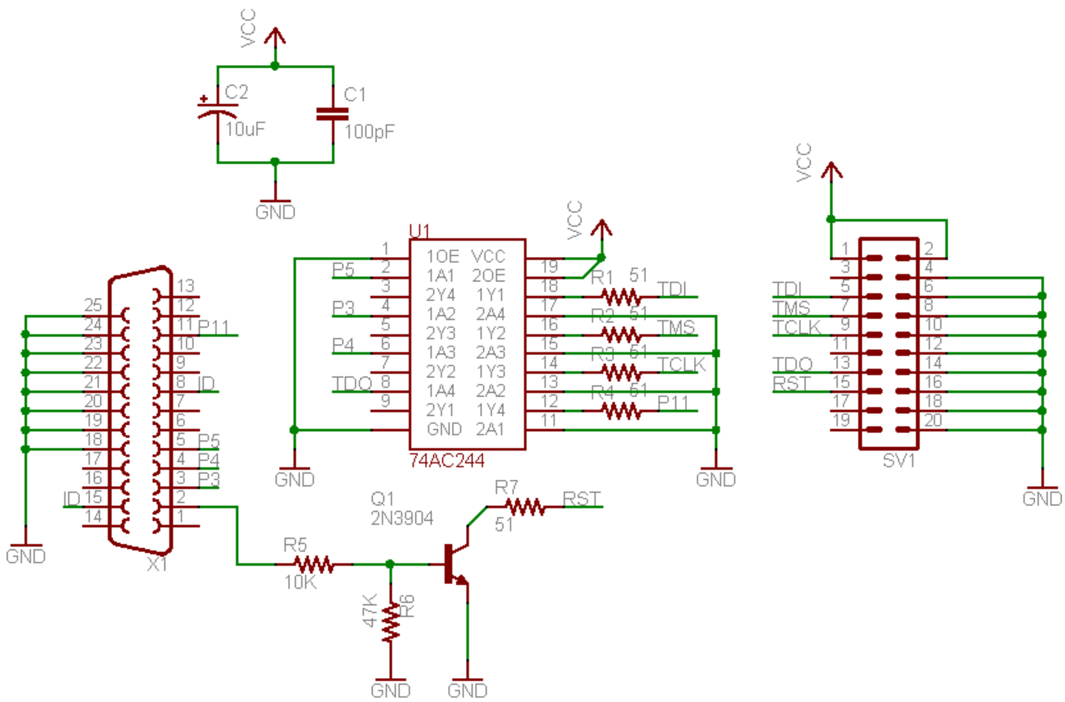
Segger

Firma Seggler udostępnia oprogramowanie komercyjne cena od 500-2000 Euro w zależności od

wersji W pracy zrezygnowano z narzędzi komercyjnych na rzecz narzędzi darmowych o publikowanych schematach i kodach źródłowych wykorzystywanych przez te programy do programowania mikrokontrolerów.

Wiggler

Dla potrzeb programowania zakupiono w firmie propox programatory korzystające z portu równoległego zgodne z Wiggler'em. Jego schemat załączony jest na rysunku poniżej:



Drugim układem do programowania pamięc i flash mikrokontrolerów wykorzystanym w pracy było rozwiązanie wykorzystujące układ ftdi. Okazało się to konieczne, albowiem w komputerach przenośnych przestaje się instalować port równoległy i oprogramowanie używające wyłącznie wiggler powoli traci zastosowanie w codziennej praktyce inżynierskiej.

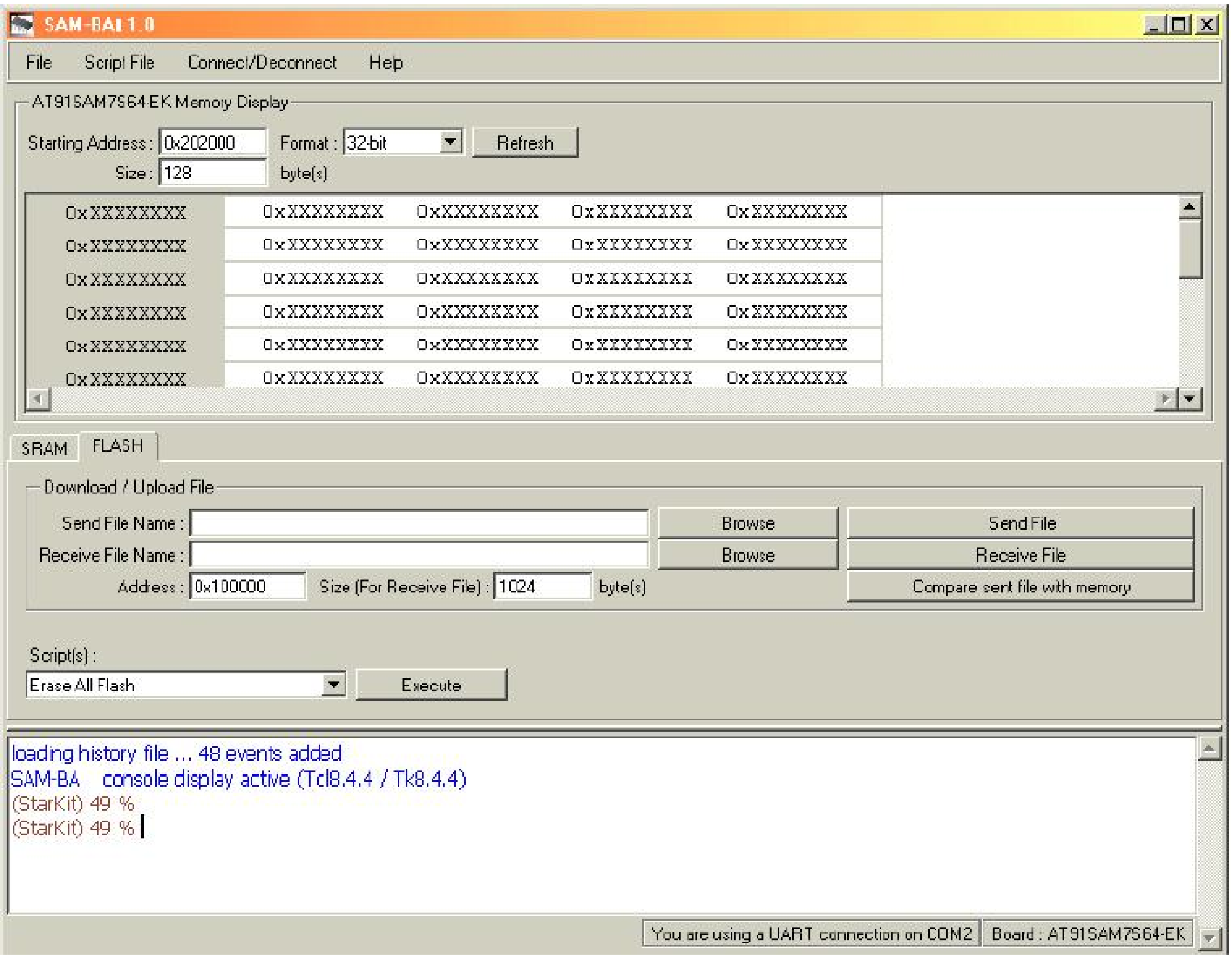
Oprogramowanie OpenOCD

Do programowania mikrokontrolerów oprócz oprócz oprogramowania dostarczonego przez producenta i zestawu ewaluacyjnego wykorzystano oprogramowanie openocd, czyli darmowy otwarty system do debugowania układów elektronicznych, system programowania i testów typu boundary-scan przygotowanym przez Dominica Rath'a. Do potrzeb pracy skompilowano wersję niezależną od programatora tj. Taką którą można używać zarówno z wigglerem i ftdi.

Dla potrzeb pracy uruchomiono i skompilowano ww. oprogramowanie ze źródeł i naniesiono odpowiednie poprawki umożliwijące używanie tego oprogramowania na platformach Linux Fedora Core 6 i OpenSuse 10.1 i 10.2. Opracowano i sprawdzono skrypty umożliwiające niezawodne programowanie używanych platform stosowanych w Z5 platform sprzętowych.

Sam-ba

At91SAM7S64, At91SAM7SA3,EVBsam7SMMsam7s256 są zestawami zawierającymi procesory ARM z corem sam7s produkowanymi przez firmę Atmel. Wspólnącechą tych układów jest możliwość programowania za pomocą interfejsu SAM-BA (SAM Boot Assistant). W przypadku procesorów AT91SAM7S64-256 samba jest na stałe umieszczona w rdzeniu procesora i umożliwia zapisanie do pamięci flash procesora dowolnego obrazu w formacie bin. Inicjalizacja samby odbywa się poprzez podanie stanu wysokiego na piny PA0,PA1,PA2 w trakcie włączenia procesora i przetrzymanie ich powyżej 5 sekund. Powoduje to zaprogramowanie flash procesora booltoaderem samba. Po ponownym włączeniu można zaprogramować mikrokontroler poprzez wejście USB z pomocą oprogramowania producenta. Odmiennie się to odbywa w przypadku procesora AT91SAM7SA3. Tam konieczne jest wgranie sam-by za pomocą wejscia dbgu. Na rysunku pod spodem pokazano główny ekran oprogramowania sam-ba:



Rysunek 4: Główne okno programu SAM-ba oprogramowanie do programowania pamięci flash mikrokontrolera

Firma Philips dostarcza swoje oprogramowanie do programowania mikrokontrolerów LPC. Procedura aktywacji oprogramowania umożliwiającego zapisanie programu do pamięci flash programowanego układu odbywa się po ustawieniu odpowiednich parametrów aplikacji i kliknięciu przycisku „Read ID”. Okno główne aplikacji „LPC2000 Flash Utility” przedstawiono poniżej:

Rysunek 5: Okno główne LPC2000 Flash Utility narzędzia do programowania pamięci flash LPC

Sprzętowe platformy systemowe

Do potrzeb pracy wykorzystano kilka platform sprzętowych pozwalających na wypróbowanie różnych systemów typu RTOS. Należały do nich:

- AT91SAM7S64-EK
- AT91SAM7SA3-EK
- EVBsam7SMMsam7s256
- EVBLPC213X
- Mmstr912
- ZL25ARM
- EP9302

Platformy AT91SAM7S64-EK,AT91SAM7SA3-EK, EVBsam7SMMsam7s256, EVBLPC213X zawierają rdzeń procesora ARM7TDMIS. Do zaportowania FreeRtos'a konieczna jest modyfikacja plików: AT91SAM7S256-RAM.ld,AT91SAM7S256-ROM.ld, interrupt_utils.c, interrupt_utils.h, simple_serial.c, simple_serial.h, startup_SAM7S.S, syscalls.c,port.c.

Platforma EP9302 posłużyła do testów, z wykorzystaniem systemu linux przeznaczonego do systemów nadzoru.

Analiza systemów RTOS o otwartym kodzie

W pracy zdecydowano się na głębszą analizę systemow typu RTOS o kodzie otwartym. Podyktowane zostało to koniecznością modyfikacji niektórych elementów systemu, w sposób umożliwiający zastosowanie ich w systemach zasilania, które w praktyce należy zaliczyć do twardych systemów czasu rzeczywistego. Niektóre z nich ze względu na rozmiar jądra nie mogą zostać wykorzystane w posiadanych platformach sprzętowych.

FreeRTOS

FreeRtos jest systemem czasu rzeczywistego przeznaczona dla systemów wbudowanych i został przystosowany do współpracy z kilkoma mikrokontrolerami. Jedo dystrybucja odbywa się na zmodyfikowanej wersji GPL. Modyfikacja licencji pozwala na tworzenie własnego zamkniętego kodu pozwlając na pozostawienie jądra systemu, jako kod o otwartym źródle. Aktualnie sytem przystosowano do pracy z procesorami o corach:

- ARM architecture ARM7
- ARM Cortex-M3
- Atmel AVR
- AVR32
- HCS12
- MicroBlaze
- MSP430
- PIC microcontroller PIC18, PIC24, dsPIC, PIC32
- Renesas H8/S
- x86
- 8052
- ColdFire (nie wspierana)

W pracy zaportowano FreeRTOS na procesory:AT91SAM7S256, AT91SAM7SA3, LPC2138, STR912.

Scheduler, czyli program szeregujący

DrRTOS

DrRtos jest bardzo prostym systemem czasu rzeczywistego. Celem tego systemu jest bycie małym, szybkim i przenośnym do wielu architektur. Autor napisał ten system ponieważ był rozczarowany innymi „darmowymi” dostępnymi systemami czasu rzeczywistego, a także został zmuszony do wypełnienia braków powołując do życia własny projekt. DrRtos posłużył mu do badań o tym jak jego system działa na różnych urządzeniach i skłonił do napisania kilku artykułów, które wyjaśniają jak działają systemy operacyjne w porównaniu do jego systemu. Zaportowany został na prosor arm core 4 i Xtensa V. Z powodu ograniczonego czasu pracy nad projektem nie zaimplementowano go na nowszy core arm.

Rodzina mikrokerneli L4

L4 jest rodziną mikrokerneli drugiej generacji opartej na projekcie i implementacji Jochana Liedtkego specjalizowanego kodu asemblerowego przeznaczonego na procesory Intel i386. Od tego czasu API uległo dramatycznemu rozwojowi w różnych kierunkach osiągając wysoki stopień niezależności i rozwojowi w zakresie bezpieczeństwa, isolacji i odporności. Nastąpiło tez kilka reimplementacji z orginalnego interfejsu jądra (ABI) i ich następców wyższego rzędu włączjąc L4Ka::Pistachio (Uniwersytet Karlsruhe), L4/MIPS (UNSW) i Fiasco (Politechnika w Dreźnie).

Z tego powodu L4 wyodrębniono jako rodzinę kerneli o różnych API.

Etapy rozwoju:

- Projekt paradygmatów
  - historyczne
    - L3
    - L4
    - L4Ka::Hazelnut
    - Fiasco
- Niezależne od platformy:
  - L4Ka::Pistachio
  - Nowsza wersja Fiasco
  - Universytet Nowej południowej Wali i NICTA

Najnowsza wersja Fiasco

Mikrokernel Fiasco jest rozwijany przez kilka lat. Obecnie wspiera kilka platform sprzętowych od x86 do AMD64, a także kilka platform z procesorem ARM. Obecnie nowa wersja Fiasco UX umożliwia uruchamianie aplikacji w trybie użytkownika na platformie Linux.

Fiasco implementuje kilka rozszerzeń do API L4v2. Rozszerzenie IPC umożliwia kernelowi przesyłanie wyjątków CPU do aplikacji użytkownika. Z wykorzystaniem tego rozszerzenia użytkownik w elegancki sposób może przejmować kontrolę nad przerwaniami systemowymi. Dodano także bloki sterujące na poziomie użytkownika w stylu X.2 (user-level thread control blocks (UTCBs)). Fiasco posiada mechanizm kontroli praw komunikacyjnych i zużycia zasobów. Na szczycie fiasco jast projektowana kolekcja podstawowych serwisów dla poziomu użytkownika (L4Env), które są wykorzystywane do para-wirtualizacji (L4Linux).

NuttX

Nuttx jest systemem czasu rzeczywistego dla urządzń wbudowanych. Główne cechy Nuttx: znajduje zastosowania miniaturowych urządzeniach osadzonych jest skalowalny od 8-mio bitowych urządzeń tiny do średnich 32-bitowych spełnia standard POSIX i Ansi, a także implementuje (fork()) typowy dla systemów głęboko osadzonych realizujący zadania w czasie rzeczywistym

Całkowicie otwarty.

Obecnie NuttX jest zaimplementowany na następujące platformy:

- TI TMS320C5471
- NXP LPC214x
- TI TMS320DM329
- PJRC 87C52

Obecna wersja /NuttX zawiera modularny mikro-kernel.

RTAI

RTAI oznacza Real-Time Application Interface, czyli interfejs aplikacji czasu-rzeczywistego. RTAI jest roższerzeniem kernela Linuksa, które pozwala na pisanie aplikacji o przewidywalnym czasie wykonania. Podobnie jak Linux RTAI jest owocem prac społeczności RTAI.

RTAI wspiera aktualnie kilka procesorów:

- x86
- x86-64
- PowerPC
- ARM (StrongARM,ARM7,CirrusLogic EP7xxx,CS89712,PXA25x)
- MIPS

RTAI umożliwia określenie deterministycznej odpowiedzi na przerwania, jest zgodny z POSIX'em i własnymi przerwaniami czasu rzeczywistego RTAI.

RTAI API zawiera dwie główne części:oparta na Adeos-ie łata do systemu, która wprowadza abstrakcyjną warstwę sprzętową szeroka gama usług która czyni programowanie systemów czasu rzeczywistego łatwiejszym.

RTLinux

RTLinux, to system czasu rzeczywistego. Profesor Victor Yodaiken oparł idee RTLinuxa na złożeniu dwóch systemów jednego z mikrokernelem działającym w czasie rzeczywistym i drugiego który nie musi być systemem czasu rzeczywistego. Problemem jest tylko wykazanie, że komponenty nie działające w czasie rzeczywistym nie zakłócają działających w czasie rzeczywistym.

Pierwsza wersja RTLinuksa ujrzała pojawiła się w 1995 roku. Aby ułatwić programistom tworzenie nowych aplikacji dla swojego środowiska oraz umożliwić korzystanie z istniejącego dorobku, RTLinux jest oczywiście zgodny z pewnymi normami, które określa profil POSIX 1003.13/PSE51(ang. Minimal Realtime Environment). RTL linux nie posiada systemu plików jako zbyt czasochłonnego w obsłudze operacji wejścia wyjścia. Umożliwia on określenie procesora na którym działa system czasu rzeczywistego, obsługa koprocesora, możliwość działania zadań w trybie periodycznym. RTLinux działa w ten sposób, że jądro „zwykłego" Linuksa jest traktowane jako zadanie i działa pod kontrolą niewielkiego i prostego systemu operacyjnego czasu rzeczywistego. Linux staje się praktycznie tzw. zadaniem tła (ang. idle task) dla RTLinuksa, które jest wykonywane tylko wtedy, kiedy nie istnieje jakiekolwiek zadanie czasu rzeczywistego ubiegające się o procesor. Założenie jest takie, że zadania Linuksa nie są w stanie zablokować przerwań ani też zapobiec wywłaszczeniu siebie. Taki cel osiągnięto dzięki implementacji programowej warstwy, która emuluje sprzętowy mechanizm kontroli przerwań. W efekcie pozbawiono „zwykłego" Linuksa możliwości blokady przerwań sprzętowych. Jeśli podejmie taką próbę, RTLinux natychmiast przechwytuje ten fakt, zaznacza odpowiednio oraz ponownie oddaje sterowanie do jądra Linuksa. Bez względu na tryb pracy, jak tryb użytkownika, systemowy czy nawet sekcja krytyczna jądra, Linux absolutnie nie jest w stanie zwiększyć czasu odpowiedzi na przerwanie czasu rzeczywistego. Jeśli ono się pojawi, jest przechwytywane przez RTLinuksa i to on decyduje co dalej z nim zrobić. Zależnie od jego natury, możliwe są dwa rozwiązania. Jeśli w systemie istnieje zarejestrowana procedura jego obsługi oraz pochodzi ona z zadania czasu rzeczywistego, to jest wywoływana. Jeśli natomiast przerwanie to ma być obsłużone przez „zwykłego" Linuksa, to wówczas jest kolejkowane i oznaczane jako oczekujące. Zostanie obsłużone dopiero, kiedy RTLinux wykryje, że jądro „dużego" systemu podjęło próbę ponownego włączenia obsługi przerwań.

Salvo

Salvo™ jest systemem czasu rzeczywistego zaprojektowanym dla zastosowań do bardzo nisko kosztowych kontrolerów o małej przestrzeni programu i danych. Dzięki niemu można szybko tworzyc aplikacje dla następujących procesorów:

- 8051 rodzina i pochodne
- ARM® ARM7TDMI® and Cortex™-M3
- Atmel® AVR® and MegaAVR™
- Motorola M68HC11
- TI MSP430 mikrokontrolery Ultra-Low Power
- Microchip PIC12|14000|16|17|18 PICmicro® MCUs
- Microchip PIC24 MCUs i dsPIC® DSCs
- Microchip PIC32™ MCUs
- TI TMS320C2000 DSPs

Wnioski

W pracy dokonano krótkiego przeglądu systemów czasu rzeczywistego z kodem otwartym, który można implementować na różnych platformach procesorowych i systemowych, a także przeglądu środowisk programistycznych i narzędzi otwartych wykorzystywanych przez programistów tworzących systemy czasu rzeczywistego. Opisano krótko modelowe platformy sprzętowe wykorzystywane w pracach Zakładu 5 Instytutu Łączności stosowane obecnie w systemach zasilania z ogniwami paliwowymi, a mający na celu szersze ich wykorzystanie w hybrydowych systemach zasilnia i nadzoru.

Do potrzeb pracy stworzono środowisko do programowania mikokontrolerów dostosowanych do potrzeb hybrydowych systemów zasilania i systemów automatycznego nadzoru. Uzyskano dzięki temu możliwość modyfikacji wszystkich narzędzi jakie są przydatne do projektowania,tworzenia, zarządzania źródłami i konserwacji oprogramowania, uniezależniając się tym samym od producentów komercyjnych kompilatorów i środowisk dewloperskich. W jego skład wchodzi pakiet narzędziowy zawierający narzędzia do edycji, kompilacji zapisu i debugowania oprogramowania, narzędzia wersjonujące i dokumentujące. Środowisko skompilowano z plików źródłowych i uruchomiono zarówno na platformie Microsoft Windows, jak i w pełni otwartej platformie Linux. Dzięki temu dokonano porównania użyteczności Windows i Linux do potrzeb profesjonalnego tworzenia kodu dla systemów czasu rzeczywistego. Platformy linuksowe ze względu na dostępność wielu narzędzi i stabilność efektów kompilacji uznano za bardziej przydatne do celów rozwoju otwartych systemów niż kłopotliwe w użyciu narzędzia adoptowane do potrzeb systemu windows.

W trakcie prac z systemem FreeRTOS, który uznano za najbardziej przydatny do potrzeb pracy zmodyfikowano odpowiednie pliki źródłowe, aby ułatwić procedury portowania, kompilacji, weryfikacji i zapisu oprogramowania do mikrokontrolerów stworzonego w projekcie. Określono poprawne przełączniki i skorygowano kod niskopoziomowy umożliwiając niezawodne funkcjonowanie adaptowanego otwartego jądra sytemu, zwracając uwagę na metody adaptacji kodu źródłowego dla potrzeb wybranych dla projektu platform sprzętowych. Napisano przykładowe programy bazujące na wybranym systemie operacyjnym, demonstrujące przydatność systemu FreeRTOS do potrzeb hybrydowych systemów zasilania i systemów automatycznego nadzoru.

Należy podkreślić badawczy charakter prac, ponieważ jest to pierwszy tego typu projekt mający na celu implementację systemu czasu rzeczywistego dla systemów zasilania, które ze względu na swoją specyfikę i niezawodność należy zaliczyć do systemów tzw. głęboko osadzonych.

Podsumowując osiągnięto cel pracy, jakim było opracowanie rdzenia systemu przeznaczonego dla systemów klasy embeded, hybrydowych systemów zasilania i systemów automatycznego nadzoru oraz dokonano niezbędnej adaptacji narzędzi softwer'owo-sprzetowych dla potrzeb programowania i analizy systemów czasu rzeczywistego.

Lista załączników

Załącznik 1:

Lista pakietów dla systemu Linux Fedora Core 6

Załącznik 2:

Lista skryptów z Listingiem ich treści umożliwiających zaprogramowanie procesorów:

At91SAM7SA3,AT91SAM7S64, AT91SAM7S256,LPC2138,STR912.

Załącznik 3:

Przykłady Programów zrealizowanych dla systemu czasu rzeczywistego FreeRTOS.

Załącznik 1

Lista Pakietów dla systemu Fedora Core 6

tzdata-2006m-2.fc6
libstdc++-4.1.1-30
expat-1.95.8-8.2.1
libacl-2.2.39-1.1
libfontenc-1.0.2-2.1
perl-5.8.8-10
ORBit2-2.14.3-3.fc6
libdv-0.104-4.fc6.1
perl-HTML-Parser-3.55-1.fc6
speex-1.0.5-2.1
file-4.17-8
libFS-1.0.0-3.1
xkeyboard-config-0.8-7.fc6
info-4.8-11.1
libsepol-1.12.27-1
libXfont-1.2.2-1.fc6
gzip-1.3.5-9
libXext-1.0.1-2.1
libXrandr-1.1.1-3.1
libXaw-1.0.2-8.1
libdmx-1.0.2-3.1
bc-1.06-21
xorg-x11-server-utils-7.1-4.fc6
fedora-release-notes-6-3
crash-4.0-3.3
festival-1.95-5.2.1
cyrus-sasl-plain-2.1.22-4
symlinks-1.2-24.2.2
setarch-2.0-1.1
rootfiles-8.1-1.1.1
krb5-libs-1.5-7
alsa-lib-1.0.12-2.fc6
GConf2-2.14.0-2.1
gtk2-engines-2.8.0-1.fc6
pwlib-1.10.1-6.fc6
dbus-0.93-3.fc6
kpartx-0.4.7-5
poppler-utils-0.5.4-2.fc6
paps-0.6.6-16.fc6
mono-data-sqlite-1.1.17.1-3.fc6
yum-3.0-6
cracklib-2.8.9-3.1
avahi-glib-0.6.11-6.fc6
kbd-1.12-18
sysklogd-1.4.1-39.2
PyXML-0.8.4-4
perl-IO-Socket-SSL-1.01-1.fc6
scrollkeeper-0.3.14-8.fc6
mkinitrd-5.1.19-1
ppp-2.4.4-1
NetworkManager-0.6.4-5.fc6
mdadm-2.5.4-2.fc6
nfs-utils-1.0.9-8.fc6
cpuspeed-1.2.1-1.40.fc6
pam_krb5-2.2.11-1
tmpwatch-2.9.7-1.1
autofs-5.0.1-0.rc2.10
krb5-workstation-1.5-7
mlocate-0.14-2.1
chkfontpath-1.10.1-1.1
gnome-desktop-2.16.0-1.fc6
eel2-2.16.0-1.fc6
gnome-python2-extras-2.14.2-4.fc6
system-config-network-1.3.95-1
gnome-python2-bonobo-2.16.0-1.fc6
beagle-0.2.10-3.fc6
sound-juicer-2.16.0-1.fc6
xorg-x11-drv-cyrix-1.1.0-4
xorg-x11-drv-savage-2.1.1-5.fc6
xorg-x11-drv-dynapro-1.1.0-2
xorg-x11-drv-mutouch-1.1.0-2
xorg-x11-drv-hyperpen-1.1.0-2
xorg-x11-drv-fbdev-0.3.0-2
xorg-x11-drv-ur98-1.1.0-1.1
zlib-devel-1.2.3-3
xml-commons-1.3.02-0.b2.7jpp.10
libbeagle-0.2.10-3.fc6
perl-BSD-Resource-1.28-1.fc6.1
libgcrypt-devel-1.2.3-1
ORBit2-devel-2.14.3-3.fc6
unixODBC-2.2.11-7.1
nss-devel-3.11.3-2
units-1.85-1.2.2
fribidi-0.10.7-5.1
cairo-java-1.0.5-3.fc6
gtkspell-2.0.11-2.1
libbonobo-devel-2.16.0-1.fc6
docbook-style-xsl-1.69.1-5.1
system-config-securitylevel-1.6.27-1
apr-util-1.2.7-3

bitmap-fonts-0.3-5.1.1
docbook-utils-pdf-0.6.14-5.1
mysql-connector-odbc-3.51.12-2.2
system-config-rootpassword-1.1.9-1
xorg-x11-apps-7.1-3.fc6
jakarta-commons-logging-javadoc-1.0.4-6jpp.1
jakarta-commons-logging-1.0.4-6jpp.1
jakarta-commons-beanutils-1.7.0-5jpp.1
mesa-libGL-devel-6.5.1-7.fc6
tomcat5-jsp-2.0-api-5.5.17-6jpp.2
kdebindings-3.5.4-1.fc6
system-config-soundcard-2.0.3-2.fc6
jakarta-commons-discovery-0.3-4jpp.1
jdepend-2.6-6jpp.1
libgnomeprint22-devel-2.12.1-8
xorg-x11-fonts-100dpi-7.1-2
ant-1.6.5-2jpp.2
libwpd-0.8.6-1
libgail-gnome-1.1.3-1.2.1
ant-nodeps-1.6.5-2jpp.2
tomcat5-server-lib-5.5.17-6jpp.2
gimp-2.2.13-1.fc6
libglade-java-2.12.5-3.fc6
openoffice.org-graphicfilter-2.0.4-5.3
ImageMagick-6.2.8.0-3.fc6
planner-0.14-3
gnome-screensaver-2.16.0-7.fc6
beagle-evolution-0.2.10-3.fc6
totem-mozplugin-2.16.1-1.fc6
synaptics-0.14.4-8.fc6
gdb-6.5-8.fc6
imake-1.0.2-3
pstack-1.2-7.2.2
redhat-rpm-config-8.0.45-6
libdrm-devel-2.0.2-1.1
strace-4.5.14-3
java_cup-manual-0.10-0.k.6jpp.1
automake-1.9.6-2.1
cyrus-sasl-devel-2.1.22-4
systemtap-0.5.9-1.fc6
pam-devel-0.99.6.2-3.fc6
java_cup-javadoc-0.10-0.k.6jpp.1
ant-javadoc-1.6.5-2jpp.2
xerces-j2-javadoc-other-2.7.1-7jpp.2
avalon-framework-javadoc-4.1.4-2jpp.13
libXi-devel-1.0.1-3.1
xmlrpc-2.0.1-3jpp.1
libXaw-devel-1.0.2-8.1
junit-demo-3.8.2-3jpp.1
libXfontcache-devel-1.0.2-3.1
hal-devel-0.5.8.1-4.fc6
kdegraphics-devel-3.5.4-1.fc6
xalan-j2-xsltc-2.7.0-6jpp.1
libgsf-devel-1.14.1-6
eclipse-jdt-3.2.1-4.fc6
bug-buddy-2.16.0-1.fc6
aspell-pl-0.51-5.2.2
mc-4.6.1a-36.20070124cvs.fc6
clips-libs-6.24-22.fc6
tolua++-1.0.92-4.fc6
clipsmm-0.0.7-1.fc6
gtkmathview-0.7.6-5.fc6
enchant-1.3.0-1.fc6
clisp-2.41-1.fc6
python-twisted-news-0.2.0-3.fc6
mikmod-3.2.2-1.fc6
python-twisted-2.4.0-3.fc6
blackbox-0.70.1-5.fc6
bzip2-1.0.3-6.fc6
clisp-devel-2.41-1.fc6
castor-demo-0.9.5-1jpp.7
bwbar-1.2.2-5
ipw3945d-1.7.22-4
wine-cms-0.9.36-2.fc6
fedora-usermgmt-default-fedora-setup-0.9-2.fc6
qps-1.9.18.6-1.fc6
moodss-21.5-1.fc6
libmad-0.15.1b-3.2.fc4.qyz
gqview-2.0.4-1.fc6
lame-3.97-1.fc6
libmms-0.3-1.fc6
aalib-1.4.0-0.11.rc5.fc6
xine-lib-moles-1.1.6-1.fc6
xmms-mp3-1.2.10-16.fc6
xmms-modplug-2.05-8.fc6
pulseaudio-lib-0.9.6-2.fc6
kaffeine-0.8.4-1.fc6
kipi-plugins-0.1.3-2.fc6
filesystem-2.4.0-1
zlib-1.2.3-3
libart_lgpl-2.3.17-4
libtiff-3.8.2-6.fc6
bluez-libs-3.7-1
libraw1394-1.2.1-1.fc6
perl-Compress-Zlib-1.42-1.fc6
pkgconfig-0.21-1.fc6
perl-IO-Zlib-1.04-4.2.1
perl-Net-IP-1.25-2.fc6
libmusicbrainz-2.1.1-4.1
pax-3.4-1.2.2

pcsc-lite-libs-1.3.1-7
mailcap-2.1.23-1.fc6
ncurses-5.5-24.20060715
nspr-4.6.3-1
iproute-2.6.16-6.fc6
libsoup-2.2.96-4.fc6
libXmu-1.0.2-5
libXxf86misc-1.0.1-3.1
procmail-3.22-17.1
giflib-4.1.3-7.1
time-1.7-27.2.2
iputils-20020927-41.fc6
xorg-x11-xauth-1.0.1-2.1
telnet-0.17-37
numactl-0.9.8-1.35
setserial-2.17-19.2.2
unix2dos-2.2-26.2.2
netpbm-10.35-6.fc6
shadow-utils-4.0.17-5
fontconfig-2.4.1-3.fc6
libXft-2.1.10-1.1
rpm-4.4.2-32
sgml-common-0.6.3-18
rpm-python-4.4.2-32
pygobject2-2.12.1-1.fc6
net-snmp-libs-5.3.1-11.fc6
bluez-pin-0.30-5
xorg-x11-utils-7.1-2.fc6
mono-core-1.1.17.1-3.fc6
yum-metadata-parser-1.0-8.fc6
pygtk2-libglade-2.10.1-4.fc6
openssh-4.3p2-10
passwd-0.73-1
dhcdbd-2.1-1.fc6
mcstrans-0.1.8-3
bind-libs-9.3.2-41.fc6
nscd-2.5-3
MAKEDEV-3.23-1.2
htmlview-3.0.0-14.1
firstboot-tui-1.4.23-1
system-config-printer-libs-0.7.32-1
apmd-3.2.2-5
irqbalance-1.13-4.fc6
vnc-server-4.1.2-3.fc6
notify-python-0.1.0-3.fc6
alsa-utils-1.0.12-3.fc6
lftp-3.5.1-2.fc6
logwatch-7.3-5
libbonoboui-2.16.0-1.fc6
ghostscript-8.15.2-8.1
librsvg2-2.16.0-2.fc6
ccid-1.0.1-5
xorg-x11-xfs-1.0.2-3.1
gthumb-2.7.8-3.fc6
gtkhtml2-2.11.0-3
gstreamer-0.10.9-2
xorg-x11-drv-vesa-1.2.1-4
xorg-x11-drv-trident-1.2.1-3.fc6
xorg-x11-drv-dmc-1.1.0-2
xorg-x11-drv-apm-1.1.1-2.1
xorg-x11-drv-palmax-1.1.0-1.1
xorg-x11-drv-jamstudio-1.1.0-1.1
xorg-x11-drv-neomagic-1.1.1-2.1
xorg-x11-drivers-7.1-3
jpackage-utils-1.6.6-1jpp.8
libXcomposite-0.3-5.1
bzip2-devel-1.0.3-3
pcre-devel-6.6-1.1
libXau-devel-1.0.1-3.1
libtiff-devel-3.8.2-6.fc6
gcc-4.1.1-30
hesiod-devel-3.1.0-8
expat-devel-1.95.8-8.2.1
gnome-audio-2.0.0-3.1.1
xorg-x11-xinit-1.0.2-12.fc6
ruby-libs-1.8.5-3.fc6
tetex-3.0-32.fc6
docbook-simple-1.0-2.1.1
mx-2.0.6-2.2.2
perl-DBI-1.52-1.fc6
guile-1.8.0-8.20060831cvs
isdn4k-utils-3.2-50
mod_python-3.2.8-3.1
cvs-1.11.22-4
xalan-j2-javadoc-2.7.0-6jpp.1
vim-enhanced-7.0.109-3
jakarta-commons-collections-3.1-6jpp.1
tomcat5-jasper-5.5.17-6jpp.2
bcel-5.1-8jpp.1
jakarta-commons-launcher-0.9-6jpp.1
java-1.4.2-gcj-compat-devel-1.4.2.0-40jpp.110
pango-devel-1.14.4-3.fc6
gjdoc-0.7.7-11
avahi-qt3-0.6.11-6.fc6
libXv-devel-1.0.1-4.1
libXxf86dga-devel-1.0.1-3.1
libswt3-gtk2-3.2.1-4.fc6
openoffice.org-core-2.0.4-5.3
ant-junit-1.6.5-2jpp.2
axis-1.2.1-2jpp.6

evolution-sharp-0.11.1-10.fc6
gimp-help-2-0.1.0.10.1.1
openoffice.org-writer-2.0.4-5.3
kdenetwork-devel-3.5.4-4.fc6
evolution-webcal-2.7.1-6
gok-1.2.0-1.fc6
gnome-python2-gtksourceview-2.16.0-1.fc6
totem-2.16.1-1.fc6
system-config-display-1.0.45-1
elfutils-libelf-devel-0.123-1.fc6
libtermcap-devel-2.0.8-46.1
bison-2.3-2.1
gmp-devel-4.1.4-7
jdepend-javadoc-2.6-6jpp.1
doxygen-1.4.7-1.1
compat-libstdc++-33-3.2.3-61
ant-manual-1.6.5-2jpp.2
libtool-1.5.22-6.1
Xaw3d-1.5E-10.1
krbafs-devel-1.2.2-10.1
automake17-1.7.9-7
puretls-javadoc-0.9-0.b5.4jpp.1
jakarta-commons-collections-javadoc-3.1-6jpp.1
jakarta-commons-beanutils-javadoc-1.7.0-5jpp.1
libXfixes-devel-4.0.1-2.1
libgnomecanvas-devel-2.14.0-4.1
libXvMC-devel-1.0.2-2.1
libXdamage-devel-1.0.3-2.1
fonts-ISO8859-2-1.0-17.1
libXTrap-devel-1.0.0-3.1
kdeutils-3.5.4-5.fc6
kdeaccessibility-3.5.4-1.fc6
gnome-vfs2-devel-2.16.0-4.fc6
frysk-0.0.1.2006.10.02.rh1-1.fc6
gnome-panel-devel-2.16.0-4.fc6
screen-4.0.2-15.1
fonts-ISO8859-2-100dpi-1.0-17.1
beryl-core-0.2.0-1.fc6
beryl-settings-0.2.0-1.fc6
python-crypto-2.0.1-4.fc6
heliodor-0.2.0-1.fc6
coldet-1.1-4.fc6
timidity++-2.13.2-1.2.2
avr-gcc-4.1.2-4.fc6
python-twisted-words-0.4.0-3.fc6
castor-0.9.5-1jpp.7
buoh-0.8.2-2.fc6
cksfv-1.3.10-1.fc6
castor-javadoc-0.9.5-1jpp.7
bubblemon-1.46-6.fc6
clearsilver-devel-0.10.4-4.fc6
cabextract-1.1-5.fc6
wine-ldap-0.9.36-2.fc6
wifi-radar-1.9.6-3.fc6
sysstat-7.0.0-1
dbmail-2.2.5-2.fc6
latex2html-2002.2.1-6
flash-plugin-9.0.48.0-release
qt-devel-3.3.8-2.fc6
libmodplug-0.8.4-1.fc6
xine-lib-1.1.6-2.fc6
mplayer-1.0-0.34.rc1try2.fc6
kdebase-devel-3.5.7-1.fc6
xosd-2.2.14-9.fc6
texi2html-1.77-0.1.20070214cvs.fc7
gxine-0.5.11-5.fc6
libsndfile-1.0.17-2.fc6
amule-2.1.3-2.fc6
setup-2.5.55-1
gnome-mime-data-2.4.2-3.1
glibc-2.5-3
chkconfig-1.3.30-1
libSM-1.0.1-3.1
libusb-0.1.12-5.1
db4-4.3.29-9.fc6
libcap-1.10-25
gnutls-1.4.1-2
elfutils-libelf-0.123-1.fc6
hesiod-3.1.0-8
perl-Digest-SHA1-2.11-1.2.1
openobex-1.3-3.1
mkisofs-2.01-10
libexif-0.6.13-2
dvd+rw-tools-6.1-4.1
perl-String-CRC32-1.4-2.fc6
perl-IO-Socket-INET6-2.51-2.fc6
libtheora-1.0alpha7-1
libmng-1.0.9-5.1
gstreamer-tools-0.10.9-2
zip-2.31-1.2.2
mingetty-1.07-5.2.2
libsysfs-2.0.0-6
libvolume_id-095-14
rmt-0.4b41-2.fc6
bash-3.1-16.1
grep-2.5.1-54.1
gawk-3.1.5-11
sqlite-3.3.6-2
audiofile-0.2.6-5
libidn-0.6.5-1.1

aspell-0.60.3-7.1
make-3.81-1.1
xorg-x11-filesystem-7.1-2.fc6
libXt-1.0.2-3.1.fc6
startup-notification-0.8-4.1
libXtst-1.0.1-3.1
libXcursor-1.1.7-1.1
gamin-0.1.7-7.fc6
libXres-1.0.1-3.1
libXxf86dga-1.0.1-3.1
iptables-ipv6-1.3.5-1.2.1
ed-0.2-38.2.2
m4-1.4.5-3
libcroco-0.6.1-2.1
unzip-5.52-2.2.1
anacron-2.3-41.fc6
iptstate-1.4-1.1.2.2
mtr-0.71-3.1
gpm-1.20.1-74.1
dump-0.4b41-2.fc6
rsync-2.6.8-3.1
hdparm-6.6-2
finger-0.17-32.2.1.1
rsh-0.17-36
rdist-6.1.5-44
brlapi-0.4.1-1.fc6
man-pages-2.39-5
libselinux-1.30.29-2
findutils-4.2.27-4.1
python-2.4.3-18.fc6
rhpl-0.194-1
pango-1.14.4-3.fc6
mesa-libGL-6.5.1-7.fc6
hicolor-icon-theme-0.9-2.1
libgnomecanvas-2.14.0-4.1
psmisc-22.2-5
gnome-icon-theme-2.16.0.1-2.fc6
pciutils-2.2.3-4
gettext-0.14.6-3.fc6
libsemanage-1.6.17-1
dbus-python-0.70-6
libgssapi-0.10-1
tar-1.15.1-19
NetworkManager-glib-0.6.4-5.fc6
opensp-1.5.2-3.1
ntsysv-1.3.30-1
pycairo-1.2.0-1.1
gtk-sharp2-2.10.0-3.fc6
gmime-sharp-2.2.3-3.fc6
python-elementtree-1.2.6-5
libselinux-python-1.30.29-2
sip-4.4.5-3
util

-linux-2.13-0.44.fc6
policycoreutils-1.30.30-1
cups-1.2.4-9
usermode-1.87-3
usermode-gtk-1.87-3
selinux-policy-2.3.18-10
bluez-utils-2.25-12
foomatic-3.0.2-38
curl-7.15.5-1.fc6
wpa_supplicant-0.4.8-10.1.fc6
tcsh-6.14-11
xml-common-0.6.3-18
lvm2-2.02.06-4
udev-095-14
kudzu-1.2.57.1-2
vim-minimal-7.0.109-3
wvdial-1.54.0-5.2.2.1
pcmciautils-014-5
nss_ldap-253-1
redhat-lsb-3.1-11
selinux-policy-targeted-2.3.18-10
setuptool-1.18.1-1.2.1
openssh-askpass-4.3p2-10
yum-updatesd-3.0-6
ipsec-tools-0.6.5-6
pam_pkcs11-0.5.3-22
libwmf-0.2.8.4-10
glx-utils-6.5.1-7.fc6
gimp-libs-2.2.13-1.fc6
sox-12.18.1-1
slrn-0.9.8.1pl1-1.2.2
wget-1.10.2-7
readahead-1.3-5
kernel-devel-2.6.18-1.2798.fc6
nss_db-2.2-35.1
libgnomeui-2.16.0-4.fc6
nautilus-extensions-2.16.0-5.fc6
libgnomeprint22-2.12.1-8
gnome-sharp-2.16.0-1.fc6
ifd-egate-0.05-15
gsf-sharp-0.8.1-2.fc6
xorg-x11-fonts-base-7.1-2
gnome-bluetooth-libs-0.7.0-10.1
sane-backends-libs-1.0.18-5.fc6
mutt-1.4.2.2-2
evince-0.6.0-3.fc6
ekiga-2.0.2-7

gucharmap-1.8.0-1.fc6
xorg-x11-drv-void-1.1.0-3.1
hplip-1.6.7-4
gnome-media-2.16.1-2.fc6
xorg-x11-server-Xorg-1.1.1-47.fc6
xorg-x11-drv-microtouch-1.1.0-1.1
xorg-x11-drv-fpit-1.1.0-1.1
xorg-x11-drv-sis-0.9.1-7
xorg-x11-drv-vmware-10.13.0-2.1
xorg-x11-drv-elographics-1.1.0-1.1
xorg-x11-drv-digitaledge-1.1.0-1.1
xorg-x11-drv-magellan-1.1.0-1.1
xorg-x11-drv-nv-1.2.0-4.fc6
xorg-x11-drv-tdfx-1.2.1-3.1
xorg-x11-drv-s3-0.4.1-2.1
xorg-x11-drv-i128-1.2.0-4
xorg-x11-drv-citron-2.2.0-1.1
xorg-x11-drv-mga-1.4.2-1.fc6
xorg-x11-drv-s3virge-1.9.1-2.1
iso-codes-0.53-1
libXdamage-1.0.3-2.1
gmp-4.1.4-7
libxklavier-3.0-1.fc6
libpng-devel-1.2.10-7
libtool-ltdl-1.5.22-6.1
libIDL-devel-0.8.7-1.fc6
perl-URI-1.35-3
libvorbis-devel-1.1.2-1.2.1
libacl-devel-2.2.39-1.1
dialog-1.0.20051107-1.2.2
glibc-devel-2.5-3
netpbm-progs-10.35-6.fc6
libgomp-4.1.1-30
libdbi-drivers-0.8.1a-1.2.2
enscript-1.6.4-4.fc6
xorg-x11-twm-1.0.1-3.1
boost-1.33.1-6.1
cdda2wav-2.01-10
xml-commons-apis-manual-1.3.02-0.b2.7jpp.10
glib-java-0.2.6-3.fc6
libxml2-devel-2.6.26-2.1.1
libgcj-devel-4.1.1-30
w3m-0.5.1-14.1
gnome-mag-0.13.1-1.fc6
passivetex-1.25-5.1.1
docbook-style-dsssl-1.79-4.1
newt-perl-1.08-9.2.2
xorg-x11-xdm-1.0.5-5.fc6
ntp-4.2.2p1-3
mod_perl-2.0.2-6.1
lm_sensors-2.10.0-3.1
dejavu-lgc-fonts-2.10-1
openssl-devel-0.9.8b-8
system-config-samba-1.2.35-1.1
mysql-server-5.0.22-2.1
crypto-utils-2.3-1
httpd-manual-2.2.3-5
xmlto-0.0.18-13.1
gnome-user-docs-2.16.0-2.fc6
a2ps-4.13b-57
freshrpms-release-1.1-1.fc
libusb-devel-0.1.12-5.1
jakarta-commons-digester-javadoc-1.7-5jpp.1
xerces-j2-javadoc-apis-2.7.1-7jpp.2
java-1.4.2-gcj-compat-1.4.2.0-40jpp.110
xml-commons-apis-1.3.02-0.b2.7jpp.10
classpathx-jaf-1.0-9jpp.1
regexp-1.4-2jpp.2
jakarta-commons-el-1.0-7jpp.1
gtksourceview-1.8.0-1.fc6
libXft-devel-2.1.10-1.1
libSM-devel-1.0.1-3.1
jakarta-oro-2.0.8-3jpp.1
kdenetwork-3.5.4-4.fc6
libXt-devel-1.0.2-3.1.fc6
gnome-doc-utils-0.8.0-2.fc6
hsqldb-1.8.0.4-3jpp.4
libXcursor-devel-1.1.7-1.1
jakarta-commons-daemon-1.0.1-6jpp.1
java-1.4.2-gcj-compat-javadoc-1.4.2.0-40jpp.110
libXdmcp-devel-1.0.1-2.1
kdesdk-3.5.4-2.fc6
xsane-0.991-2.fc6
xorg-x11-fonts-ISO8859-1-100dpi-7.1-2
java_cup-0.10-0.k.6jpp.1
log4j-1.2.13-3jpp.2
eclipse-rcp-3.2.1-4.fc6
ant-trax-1.6.5-2jpp.2
nautilus-sendto-0.7-5.fc6
ant-commons-logging-1.6.5-2jpp.2
ant-apache-bcel-1.6.5-2jpp.2
ant-jdepend-1.6.5-2jpp.2
mx4j-3.0.1-6jpp.4
geronimo-specs-compat-1.0-0.M2.2jpp.12
gcalctool-5.8.24-2.fc6
kdnssd-avahi-devel-0.1.3-0.1.20060713svn.fc6
gimp-print-plugin-4.2.7-22
eclipse-platform-3.2.1-4.fc6
openoffice.org-math-2.0.4-5.3
openoffice.org-xsltfilter-2.0.4-5.3

gnome-games-2.16.0-1.fc6
devhelp-0.12-6.fc6
gaim-2.0.0-0.15.beta3.fc6
vino-2.13.5-4.1
desktop-printing-0.19-16.fc6
gnome-python2-gconf-2.16.0-1.fc6
system-config-date-1.8.7-1
pirut-1.2.5-1
gnome-volume-manager-2.15.0-2.fc6
gnome-applets-2.16.0.1-7.fc6
compiz-0.0.13-0.32.20060817git.fc6
xorg-x11-util-macros-1.0.2-4.fc6
libstdc++-devel-4.1.1-30
libXvMC-1.0.2-2.1
readline-devel-5.1-1.1
gcc-c++-4.1.1-30
gpm-devel-1.20.1-74.1
xorg-x11-xtrans-devel-1.0.1-1.1.fc6
gdbm-devel-1.8.0-26.2.1
automake16-1.6.3-8
netpbm-devel-10.35-6.fc6
ltrace-0.5-6.45svn.fc6
autorun-3.20-1.1
byacc-1.9-29.2.2
log4j-manual-1.2.13-3jpp.2
xalan-j2-manual-2.7.0-6jpp.1
avalon-framework-manual-4.1.4-2jpp.13
GConf2-devel-2.14.0-2.1
sip-devel-4.4.5-3
gnome-keyring-devel-0.6.0-1.fc6
libvte-java-0.12.1-4.fc6
libselinux-devel-1.30.29-2
curl-devel-7.15.5-1.fc6
subversion-1.3.2-6
lockdev-devel-1.0.1-10
jakarta-taglibs-standard-javadoc-1.1.1-7jpp.1
cryptix-javadoc-3.2.0-9jpp.1
xerces-j2-javadoc-impl-2.7.1-7jpp.2
jakarta-commons-lang-javadoc-2.1-5jpp.1
antlr-javadoc-2.7.6-4jpp.2
jakarta-commons-dbcp-javadoc-1.2.1-7jpp.1
jakarta-commons-modeler-javadoc-1.1-8jpp.1
libXpm-devel-3.5.5-3
pycairo-devel-1.2.0-1.1
libglade2-devel-2.6.0-2
gnu-crypto-2.1.0-2jpp.1
at-spi-devel-1.7.11-2.fc6
jdepend-demo-2.6-6jpp.1
jakarta-commons-validator-1.1.4-5jpp.1
rpm-devel-4.4.2-32
libXScrnSaver-devel-1.1.0-3.1
libXxf86vm-devel-1.0.1-3.1
libXxf86misc-devel-1.0.1-3.1
java-1.4.2-gcj-compat-src-1.4.2.0-40jpp.110
avalon-logkit-1.2-4jpp.3
kdeartwork-3.5.4-1.fc6
ant-scripts-1.6.5-2jpp.2
kdbg-2.0.2-1.2.1
libgnomeui-devel-2.16.0-4.fc6
eclipse-changelog-2.3.2-1.fc6
gnome-pilot-devel-2.0.13-16
evolution-data-server-devel-1.8.0-11.fc6
vnc-4.1.2-3.fc6
zsh-4.2.6-1
man-pages-pl-0.24-2.1
kyum-0.7.5-4.fc6
cairo-1.2.6-1.fc6
bzip2-libs-1.0.3-6.fc6
beryl-plugins-0.2.0-1.fc6
beryl-settings-simple-0.2.0-1.fc6
gtk+-1.2.10-55.fc6
libsigc++20-devel-2.0.17-2
beryl-kde-0.2.0-1.fc6
gtkglext-1.2.0-4.fc6
ots-0.4.2-10.fc6
tk-8.4.13-3.fc6
cegui-0.4.1-11.fc6
avr-binutils-2.17-3.fc6
python-twisted-names-0.3.0-3.fc6
python-twisted-runner-0.2.0-4.fc6
glibmm24-devel-2.12.8-1.fc6
adaptx-0.9.13-3jpp.1
python-twisted-web-0.6.0-4.fc6
ogre-1.2.2-2.p1.fc6
clipsmm-devel-0.0.7-1.fc6
beryl-core-devel-0.2.0-1.fc6
autoconf213-2.13-12.1
castor-xml-0.9.5-1jpp.7
castor-test-0.9.5-1jpp.7
castor-doc-0.9.5-1jpp.7
cairo-devel-1.2.6-1.fc6
bsh-javadoc-1.3.0-9jpp.1
celestia-1.4.1-7.fc6
cfv-1.18.1-2.fc6
blt-2.4-14.z.fc6
dkms-2.0.13-1.fc6
wine-twain-0.9.36-2.fc6
wine-esd-0.9.36-2.fc6
wine-0.9.36-2.fc6
libsieve-2.2.5-1.fc6

vnstat-1.4-8.fc6
fedora-usermgmt-core-0.9-2.fc6
sqlite2-2.8.17-1.fc6
kismet-extras-0.0.2006.04.R1-4.fc6
libid3tag-0.15.1b-3.qyz
libao-0.8.6-5.qyz
glib-1.2.10-26
arts-1.5.7-0.1.fc6
libmpcdec-1.2.2-4.fc6
xvidcore-1.1.3-2.fc6
freeglut-2.4.0-11.fc6
mcs-libs-0.4.1-3.fc6
libmp4v2-1.5.0.1-3.fc6
kdelibs-devel-3.5.7-1.fc6
libfame-0.9.1-12.fc6
qt-designer-3.3.8-2.fc6
akode-2.0.1-5.fc6
xmms-wma-1.0.5-3.fc6
xmms-xosd-2.2.14-9.fc6
qemu-0.8.2-4
libfreebob-1.0.0-3.fc6
xine-plugin-1.0-3.fc6
gxine-mozplugin-0.5.11-5.fc6
k3b-0.12.17-1
mplayer-skins-1.8-1
rar-3.7.0-0.1.beta1.rh9.rf
libgcc-4.1.1-30
kernel-headers-2.6.18-1.2798.fc6
glibc-common-2.5-3
atk-1.12.2-1.fc6
libICE-1.0.1-2.1
libjpeg-6b-37
audit-libs-1.2.8-1.fc6
cyrus-sasl-lib-2.1.22-4
libogg-1.1.3-2.fc6
libgcrypt-1.2.3-1
tcp_wrappers-7.6-40.2.1
libXdmcp-1.0.1-2.1
slang-2.0.6-3
libvorbis-1.1.2-1.2.1
wireless-tools-28-1.fc6
libdrm-2.0.2-1.1
liboil-0.3.8-2.1
perl-Digest-HMAC-1.01-15
perl-Socket6-0.19-3.fc6
libiec61883-1.0.0-11.fc6
lcms-1.15-1.2.2
cdrdao-1.2.1-2
dosfstools-2.11-6.1.fc6
ethtool-3-1.2.2
libevent-1.1a-3.2.1
libdaemon-0.10-3.2
nash-5.1.19-1
libtermcap-2.0.8-46.1
libbonobo-2.16.0-1.fc6
readline-5.1-1.1
desktop-file-utils-0.10-7
diffutils-2.8.1-15.2.2
procps-3.2.7-8
pilot-link-0.11.8-16
cpio-2.6-19
fedora-logos-6.0.6-1.fc6
libXrender-0.9.1-3.1
libXi-1.0.1-3.1
libXxf86vm-1.0.1-3.1
libXfixes-4.0.1-2.1
cdparanoia-libs-alpha9.8-27.2
libXTrap-1.0.0-3.1
libXfontcache-1.0.2-3.1
xorg-x11-xkb-utils-1.0.2-2.1
ttmkfdir-3.0.9-22
groff-1.18.1.1-11.1
shared-mime-info-0.19-1
fedora-release-6-4
aspell-en-6.0-2.1
nano-1.3.12-1.1
mgetty-1.1.33-9.fc6
nc-1.84-10.fc6
acl-2.2.39-1.1
vconfig-1.9-2.1
pam_smb-1.1.7-7.2.1
pam_passwdqc-1.0.2-1.2.2
eject-2.1.5-4
specspo-12-1
gnome-backgrounds-2.15.92-1.fc6
e2fsprogs-libs-1.39-7
openssl-0.9.8b-8
openldap-2.3.27-4
newt-0.52.2-9
hwdata-0.191-1
gtk2-2.10.4-4.fc6
gnome-keyring-0.6.0-1.fc6
net-tools-1.60-73
poppler-0.5.4-2.fc6
mesa-libGLU-6.5.1-7.fc6
libbtctl-0.6.0-9.1
libxml2-python-2.6.26-2.1.1
notification-daemon-0.3.5-6
e2fsprogs-1.39-7
dmraid-1.0.0.rc13-1.fc6

opal-2.2.2-1.1
openjade-1.3.2-27
system-config-securitylevel-tui-1.6.27-1
gd-2.0.33-9.3.fc6
mono-data-1.1.17.1-3.fc6
pyxf86config-0.3.31-2.fc6
python-sqlite-1.1.7-1.2.1
python-numeric-23.7-2.2.2
PyQt-3.16-4
SysVinit-2.86-14
avahi-0.6.11-6.fc6
portmap-4.0-65.2.2.1
authconfig-5.3.10-1
ypbind-1.19-5
openssh-clients-4.3p2-10
cyrus-sasl-2.1.22-4
at-3.1.8-82.fc6
libwvstreams-4.2.2-2.1
perl-Net-SSLeay-1.30-4.fc6
mtools-3.9.10-2.fc6
docbook-dtds-1.0-30.1
lockdev-1.0.1-10
hal-0.5.8.1-4.fc6
system-config-network-tui-1.3.95-1
logrotate-3.7.4-7
sudo-1.6.8p12-10
mkbootdisk-1.5.3-2.1
bind-utils-9.3.2-41.fc6
fetchmail-6.3.4-1.1
yp-tools-2.9-0.1
system-config-keyboard-1.2.10-2
quota-3.13-1.2.3.1
microcode_ctl-1.13-1.33.fc6
rng-utils-2.0-1.14.1.fc6
pam_ccreds-3-5
xterm-215-3.fc6
gnome-themes-2.16.0-1.fc6
cadaver-0.22.3-4
irda-utils-0.9.17-2.fc6
elinks-0.11.1-5
tcpdump-3.9.4-8.1
ksh-20060214-1.1
parted-1.7.1-16.fc6
gnome-vfs2-2.16.0-4.fc6
gnome-python2-2.16.0-1.fc6
pcsc-lite-1.3.1-7
libgnomeprintui22-2.12.1-6
gnome-mount-0.5-2.fc6
xorg-x11-drv-keyboard-1.1.0-2.1
gnome-pilot-2.0.13-16
gnome-python2-canvas-2.16.0-1.fc6
gnome-spell-1.0.7-3.1
gnome-bluetooth-0.7.0-10.1
coolkey-1.0.1-10
file-roller-2.16.0-2.fc6
gnome-python2-desktop-2.16.0-1.fc6
libsane-hpaio-1.6.7-4
beagle-gui-0.2.10-3.fc6
xorg-x11-drv-evdev-1.1.2-2.1
xorg-x11-drv-mouse-1.1.1-1.1
xorg-x11-drv-i740-1.1.0-2.1
xorg-x11-drv-nsc-2.8.1-2.1
xorg-x11-drv-sisusb-0.8.1-4.1
xorg-x11-drv-v4l-0.1.1-4
xorg-x11-drv-ark-0.6.0-2.1
xorg-x11-drv-via-0.2.1-7
xorg-x11-drv-ast-0.81.0-3
xorg-x11-drv-rendition-4.1.0-3.1
xorg-x11-drv-magictouch-1.0.0.5-2.1
xorg-x11-drv-voodoo-1.1.0-3.1
xorg-x11-drv-i810-1.6.5-9.fc6
xorg-x11-drv-ati-6.6.2-4.fc6
xorg-x11-drv-vmmouse-12.4.0-2.1
xorg-x11-drv-elo2300-1.1.0-1.1
comps-extras-11.1-1.1
freetype-devel-2.2.1-10.fc6
libjpeg-devel-6

b-37
psutils-1.17-26.1
tetex-dvips-3.0-32.fc6
libart_lgpl-devel-2.3.17-4
libmng-devel-1.0.9-5.1
libidn-devel-0.6.5-1.1
libogg-devel-1.1.3-2.fc6
libattr-devel-2.4.32-1.1
vim-common-7.0.109-3
glibc-headers-2.5-3
libgpod-0.3.0-3.1
libpfm-3.2-0.060621.8.1
libdbi-0.8.1-2.1
gnuplot-4.0.0-12
elfutils-libs-0.123-1.fc6
bitstream-vera-fonts-1.10-7
lrzsz-0.12.20-22.1
kudzu-devel-1.2.57.1-2
at-spi-1.7.11-2.fc6
libgtk-java-2.8.6-4.fc6
alsa-lib-devel-1.0.12-2.fc6
esound-devel-0.2.36-3
libxslt-devel-1.1.17-1.1

xmltex-20020625-8
zenity-2.16.0-1.fc6
jadetex-3.12-13.1.1
samba-3.0.23c-2
htdig-3.2.0b6-6.4.3
httpd-2.2.3-5
perl-DBD-MySQL-3.0007-1.fc6
vte-0.14.0-1.fc6
krb5-devel-1.5-7
system-config-services-0.9.1-1.fc6
php-ldap-5.1.6-3
libdbi-dbd-mysql-0.8.1a-1.2.2
webalizer-2.01_10-30.1
samba-client-3.0.23c-2
docbook-slides-3.3.1-2.1.1
xhtml1-dtds-1.0-7.1.1
vorbis-tools-1.1.1-2
gpg-pubkey-e42d547b-3960bdf1
libao-devel-0.8.6-5.qyz
gdk-pixbuf-0.22.0-32.fc6
tux-3.2.18-9.fc6
jakarta-commons-daemon-javadoc-1.0.1-6jpp.1
xml-commons-which-javadoc-1.3.02-0.b2.7jpp.10
xorg-x11-proto-devel-7.1-9.fc6
classpathx-mail-1.1.1-4jpp.2
jakarta-commons-digester-1.7-5jpp.1
jakarta-commons-pool-1.3-5jpp.1
dbus-x11-0.93-3.fc6
jakarta-commons-fileupload-1.0-6jpp.1
eclipse-ecj-3.2.1-4.fc6
kdepim-3.5.4-4.fc6
hal-cups-utils-0.6.2-4
mesa-libGLU-devel-6.5.1-7.fc6
jakarta-commons-httpclient-3.0-7jpp.1
libXinerama-devel-1.0.1-2.1
junit-3.8.2-3jpp.1
jzlib-1.0.7-4jpp.1
jakarta-commons-pool-javadoc-1.3-5jpp.1
gnome-user-share-0.10-5
xorg-x11-fonts-misc-7.1-2
xorg-x11-fonts-Type1-7.1-2
xerces-j2-2.7.1-7jpp.2
xml-commons-resolver-1.1-1jpp.12
ant-apache-resolver-1.6.5-2jpp.2
bsf-2.3.0-11jpp.1
gnome-netstatus-2.12.0-5.1
ant-antlr-1.6.5-2jpp.2
ant-swing-1.6.5-2jpp.2
ant-jsch-1.6.5-2jpp.2
jakarta-commons-modeler-1.1-8jpp.1
tomcat5-common-lib-5.5.17-6jpp.2
libgnome-java-2.12.4-3.fc6
nautilus-2.16.0-5.fc6
gimp-data-extras-2.0.1-1.1.1
nautilus-sendto-bluetooth-0.7-5.fc6
openoffice.org-calc-2.0.4-5.3
sane-frontends-1.0.14-1.2.2
gdm-2.16.0-10.fc6
tomboy-0.4.1-1.fc6
eog-2.16.0.1-2.fc6
gnome-power-manager-2.16.0-3.fc6
im-chooser-0.3.3-2.fc6
gnome-python2-applet-2.16.0-1.fc6
gedit-2.15.9-1.fc6
system-config-lvm-1.0.18-1.2.FC6
gnome-session-2.16.0-3.fc6
orca-1.0.0-4.fc6
firstboot-1.4.23-1
elfutils-0.123-1.fc6
libgfortran-4.1.1-30
libsepol-devel-1.12.27-1
pilot-link-devel-0.11.8-16
gcc-gfortran-4.1.1-30
xrestop-0.2-6.2.2
ncurses-devel-5.5-24.20060715
texinfo-4.8-11.1
automake14-1.4p6-13
pcsc-lite-devel-1.3.1-7
db4-devel-4.3.29-9.fc6
xdelta-1.1.3-20
rcs-5.7-30.1
junit-manual-3.8.2-3jpp.1
cryptix-asn1-javadoc-20011119-7jpp.2
xorg-x11-docs-1.2-4.fc6
python-devel-2.4.3-18.fc6
libgconf-java-2.12.4-4.fc6
krbafs-1.2.2-10.1
oprofile-0.9.1-16
gtk-doc-1.7-1.fc6
valgrind-3.2.1-4
PyQt-devel-3.16-4
xml-commons-which-1.3.02-0.b2.7jpp.10
jakarta-commons-validator-javadoc-1.1.4-5jpp.1
regexp-javadoc-1.4-2jpp.2
xml-commons-apis-javadoc-1.3.02-0.b2.7jpp.10
bcel-javadoc-5.1-8jpp.1
avalon-logkit-javadoc-1.2-4jpp.3
classpathx-jaf-javadoc-1.0-9jpp.1
xerces-j2-javadoc-xni-2.7.1-7jpp.2

gtk2-devel-2.10.4-4.fc6
jlex-1.2.6-5jpp.1
jessie-1.0.1-7
pygtk2-devel-2.10.1-4.fc6
Xaw3d-devel-1.5E-10.1
gd-devel-2.0.33-9.3.fc6
libXevie-devel-1.0.1-3.1
libXres-devel-1.0.1-3.1
libXfont-devel-1.2.2-1.fc6
jakarta-commons-lang-2.1-5jpp.1
kdegraphics-3.5.4-1.fc6
avalon-framework-4.1.4-2jpp.13
libgnomeprintui22-devel-2.12.1-6
xalan-j2-demo-2.7.0-6jpp.1
xerces-j2-scripts-2.7.1-7jpp.2
libbonoboui-devel-2.16.0-1.fc6
librsvg2-devel-2.16.0-2.fc6
eel2-devel-2.16.0-1.fc6
eclipse-bugzilla-0.2.4-3.fc6
firefox-devel-1.5.0.7-7.fc6
glade2-2.12.1-5.fc6
nmap-4.11-1.1
zisofs-tools-1.0.6-3.2.2
gpg-pubkey-1ac70ce6-41bebeef
minicom-2.2-1.fc6
glibmm24-2.12.8-1.fc6
emerald-0.2.0-1.fc6
gtkmm24-2.10.9-1.fc6
beryl-manager-0.2.0-1.fc6
bittorrent-4.4.0-5.fc6
clips-devel-6.24-22.fc6
aquamarine-0.2.0-1.fc6
goffice-0.2.1-2.fc6
pyOpenSSL-0.6-1.p24.7.2.2
tcl-8.4.13-3.fc6
DevIL-1.6.8-0.11.rc2.fc6
mathml-fonts-1.0-21.fc6
python-twisted-core-2.4.0-6.fc6
python-twisted-conch-0.7.0-4.fc6
perl-XML-Parser-2.34-6.1.2.2.1
ClanLib06-0.6.5-7.fc6
SOAPpy-0.11.6-5.fc6
zziplib-0.13.49-1.fc6
bison-runtime-2.3-2.1
bluefish-1.0.7-1.fc6
clanbomber-1.05-3.fc6
clearsilver-0.10.4-4.fc6
clips-xclips-6.24-22.fc6
chess-1.0-3.fc6
bwm-ng-0.5-8.fc6
clips-6.24-22.fc6
bsh-manual-1.3.0-9jpp.1
beryl-0.2.0-1.fc6
bsh-demo-1.3.0-9jpp.1
dkms-ipw3945-1.2.1-1
wine-capi-0.9.36-2.fc6
wine-nas-0.9.36-2.fc6
jre-1.6.0_01-fcs
tclx-8.4.0-5.fc6
knetstats-1.6.1-2.fc6
fedora-usermgmt-shadow-utils-0.9-2.fc6
sqlite2-tcl-2.8.17-1.fc6
moomps-5.8-2.fc6
libid3tag-devel-0.15.1b-3.qyz
glib-devel-1.2.10-26
kdelibs-3.5.7-1.fc6
lirc-0.8.1-1.fc6
x264-0.0.0-0.3.20061214.fc6
libcaca-0.99-0.1.beta11.fc6
audacious-libs-1.3.2-1.fc6
faac-1.25-2.fc6
xmms-libs-1.2.10-29.fc6
libdvdcss-1.2.9-2.fc6
audacious-plugins-extras-1.3.3-1.fc6
mencoder-1.0-0.34.rc1try2.fc6
xmms-cdread-0.14-12.fc6
xmms-musepack-1.2-3.fc6
ktorrent-2.2.2-1.fc6
jack-audio-connection-kit-0.103.0-1.fc6
xine-skins-1.10-1.fc
imlib2-1.3.0-3.fc6
libkipi-0.1.5-1.fc6
yumex-2.0.1-1.fc6
basesystem-8.0-5.1.1
glib2-2.12.3-2.fc6
mktemp-1.5-23.2.2
libXau-1.0.1-3.1
libgpg-error-1.4-2
beecrypt-4.1.2-10.1.1
libavc1394-0.5.3-1.fc6
SDL-1.2.10-6.2
perl-Archive-Tar-1.30-1.fc6
perl-Net-DNS-0.59-1.fc6
pcre-6.6-1.1
mailx-8.1.1-44.2.2
libnl-1.0-0.10.pre5.4
termcap-5.5-1.20060701.1
freetype-2.2.1-10.fc6
nss-3.11.3-2
iptables-1.3.5-1.2.1

libxslt-1.1.17-1.1
libXinerama-1.0.1-2.1
libXv-1.0.1-4.1
cdparanoia-alpha9.8-27.2
libxkbfile-1.0.3-3.1
crontabs-1.10-8
grub-0.97-13
talk-0.17-29.2.2
fbset-2.1-22
traceroute-1.0.4-2
tree-1.5.0-4
desktop-backgrounds-basic-2.0-37
device-mapper-1.02.07-3
neon-0.25.5-5.1
gail-1.9.2-1.fc6
rpm-libs-4.4.2-32
gnome-menus-2.16.0-2.fc6
libnotify-0.4.2-4.fc6
nfs-utils-lib-1.0.8-7.2
man-1.6d-1.1
libgdiplus-1.1.17-1.fc6
python-urlgrabber-2.9.9-2
pygtk2-2.10.1-4.fc6
initscripts-8.45.3-1
libuser-0.54.7-2
libgnomecups-0.2.2-8
sendmail-8.13.8-2
libpcap-0.9.4-8.1
syslinux-3.11-4
gphoto2-2.2.0-2.1
which-2.16-7
pinfo-0.6.9-1.fc6
gnupg-1.4.5-4
authconfig-gtk-5.3.10-1
acpid-1.0.4-5
prelink-0.3.9-2
device-mapper-multipath-0.4.7-5
usbutils-0.71-2.1
stunnel-4.15-2
sysreport-1.4.3-9
libgnome-2.16.0-4.fc6
urw-fonts-2.3-6.1.1
libgsf-1.14.1-6
gtkhtml3-3.12.0-1.fc6
gnome-python2-gnomevfs-2.16.0-1.fc6
evolution-2.8.0-7.fc6
gimp-print-4.2.7-22
sane-backends-1.0.18-5.fc6
gstreamer-plugins-base-0.10.9-4
xorg-x11-drv-siliconmotion-1.4.1-2.1
xorg-x11-drv-penmount-1.1.0-2.1
xorg-x11-drv-cirrus-1.1.0-2.fc6
xorg-x11-drv-aiptek-1.0.1-2
xorg-x11-drv-vga-4.1.0-2.1
xorg-x11-drv-glint-1.1.1-4.1
xorg-x11-drv-chips-1.1.1-2.1
php-common-5.1.6-3
glib2-devel-2.12.3-2.fc6
tetex-fonts-3.0-32.fc6
agg-2.4-2.1
libXevie-1.0.1-3.1
nspr-devel-4.6.3-1
libicu-3.6-4
libsilc-1.0.2-2.fc6
dcraw-0.0.20060521-1.1
joystick-1.2.15-20.2.2
libgcj-4.1.1-30
apr-1.2.7-10
ruby-1.8.5-3.fc6
tetex-latex-3.0-32.fc6
perl-SGMLSpm-1.03ii-16.2.1
distcache-1.4.5-14.1
mysql-5.0.22-2.1
e2fsprogs-devel-1.39-7
rhgb-0.16.4-1.fc6
mod_ssl-2.2.3-5
squid-2.6.STABLE4-1.fc6
gnu-crypto-javadoc-2.1.0-2jpp.1
jakarta-commons-fileupload-javadoc-1.0-6jpp.1
libX11-devel-1.0.3-4.fc6
libXext-devel-1.0.1-2.1
jakarta-commons-dbcp-1.2.1-7jpp.1
antlr-2.7.6-4jpp.2
ldapjdk-4.17-1jpp.7
jsch-0.1.28-1jpp.5
gimp-print-utils-4.2.7-22
xorg-x11-fonts-truetype-7.1-2
bsh-1.3.0-9jpp.1
ant-apache-oro-1.6.5-2jpp.2
wsdl4j-1.5.2-4jpp.1
tomcat5-5.5.17-6jpp.2
xsane-gimp-0.991-2.fc6
openoffice.org-impress-2.0.4-5.3
kdepim-devel-3.5.4-4.fc6
NetworkManager-gnome-0.6.4-5.fc6
gnome-system-monitor-2.16.0-1.fc6
gnome-python2-gnomeprint-2.16.0-1.fc6
control-center-2.16.0-9.fc6
rhpxl-0.39-2
slang-devel-2.0.6-3

libfontenc-devel-1.0.2-2.1
giflib-devel-4.1.3-7.1
cscope-15.5-15.fc6.1
automake15-1.5-16
swig-1.3.29-1.fc6
compat-libstdc++-296-2.96-138
ldapjdk-javadoc-4.17-1jpp.7
antlr-manual-2.7.6-4jpp.2
pygobject2-devel-2.12.1-1.fc6
neon-devel-0.25.5-5.1
openldap-devel-2.3.27-4
newt-devel-0.52.2-9
jakarta-commons-launcher-javadoc-0.9-6jpp.1
jakarta-oro-javadoc-2.0.8-3jpp.1
log4j-javadoc-1.2.13-3jpp.2
libXmu-devel-1.0.2-5
gail-devel-1.9.2-1.fc6
coolkey-devel-1.0.1-10
libXcomposite-devel-0.3-5.1
fonts-ISO8859-2-75dpi-1.0-17.1
jakarta-commons-el-javadoc-1.0-7jpp.1
dbus-glib-devel-0.70-4
jakarta-taglibs-standard-1.1.1-7jpp.1
libgnome-devel-2.16.0-4.fc6
kdeutils-devel-3.5.4-5.fc6
gnome-desktop-devel-2.16.0-1.fc6
openldap-clients-2.3.27-4
openoffice.org-langpack-pl_PL-2.0.4-5.3
libsigc++20-2.0.17-2
cairomm-1.2.4-1.fc6
Hermes-1.3.3-12.fc6
beryl-gnome-0.2.0-1.fc6
c-ares-1.3.2-1.fc6
allegro-4.2.0-18.fc6
python-zope-interface-3.0.1-6.fc6
perl-libwww-perl-5.805-1.1.1
python-fpconst-0.7.2-3.fc6
bsd-games-2.17-17.fc6
clips-doc-6.24-22.fc6
bzflag-2.0.8-3.fc6
avr-gcc-c++-4.1.2-4.fc6
avrdude-5.3.1-5.fc6
chromium-0.9.12-21.qyz
wine-core-0.9.36-2.fc6
wine-tools-0.9.36-2.fc6
tktable-2.9-9.fc6
fedora-usermgmt-0.9-2.fc6
texinfo-tex-4.8-11.1
mpg321-0.2.10.3-3.qyz
kdebase-3.5.7-1.fc6
kdemultimedia-3.5.7-1.fc6
wavpack-4.41-1.fc6
mplayer-fonts-1.1-3.fc
taglib-1.4-5.fc6
xmms-1.2.10-29.fc6
compat-gcc-34-3.4.6-7
js-1.60-2.fc6
exiv2-0.12-1.fc6
wxGTK-2.6.3-2.6.3.2.3.fc6
cracklib-dicts-2.8.9-3.1
popt-1.10.2-32
libpng-1.2.10-7
libattr-2.4.32-1.1
cups-libs-1.2.4-9
gdbm-1.8.0-26.2.1
libIDL-0.8.7-1.fc6
libieee1284-0.2.9-3.2.2
perl-HTML-Tagset-3.10-2.1.1
flac-1.1.2-27
gmime-2.2.3-3.fc6
patch-2.5.4-29.2.2
dmidecode-2.7-1.26.1.fc6
libxml2-2.6.26-2.1.1
sed-4.1.5-5.fc6
redhat-menus-6.7.5-3
less-394-4.1
libX11-1.0.3-4.fc6
xorg-x11-font-utils-7.1-2
libXpm-3.5.5-3
libXScrnSaver-1.1.0-3.1
nss-tools-3.11.3-2
cpp-4.1.1-30
cdrecord-2.01-10
ftp-0.17-33.fc6
jwhois-3.2.3-6.1
attr-2.4.32-1.1
rdate-1.4-6
dos2unix-3.1-27.1
words-3.0-9
coreutils-5.97-11
module-init-tools-3.3-0.pre1.4.17
esound-0.2.36-3
libglade2-2.6.0-2
libwnck-2.16.0-4.fc6
ghostscript-fonts-5.50-13.1.1
dbus-glib-0.70-4
libutempter-1.1.4-3.fc6
redhat-artwork-5.0.8-1.fc6
mono-web-1.1.17.1-3.fc6
audit-libs-python-1.2.8-1.fc6

pam-0.99.6.2-3.fc6
dhclient-3.0.4-21.fc6
pm-utils-0.19-3
vixie-cron-4.1-64.fc6
pyorbit-2.14.1-1.1
spamassassin-3.1.4-1.fc6
cryptsetup-luks-1.0.3-2.

1
kernel-2.6.18-1.2798.fc6
rp-pppoe-3.5-32.1
smartmontools-5.36-3
netdump-0.7.16-5
openssh-server-4.3p2-10
dhcpv6_client-0.10-32.fc6
samba-common-3.0.23c-2
xsri-2.1.0-10.fc6
diskdumputils-1.3.6-5.1
psacct-6.3.2-41.1
lsof-4.78-3
evolution-data-server-1.8.0-11.fc6
gnome-panel-2.16.0-4.fc6
nautilus-cd-burner-2.16.0-3.fc6
firefox-1.5.0.7-7.fc6
gnome-python2-libegg-2.14.2-4.fc6
xorg-x11-fonts-ISO8859-1-75dpi-7.1-2
hpijs-1.6.7-4
gstreamer-plugins-good-0.10.4-1.fc6
xorg-x11-drv-spaceorb-1.1.0-1.1
xorg-x11-drv-summa-1.1.0-1.1
xorg-x11-drv-acecad-1.1.0-2.1
xorg-x11-drv-joystick-1.1.0-1.1
xorg-x11-drv-dummy-0.2.0-2.1
xorg-x11-drv-calcomp-1.1.0-1.1
xorg-x11-drv-tseng-1.1.0-3.1
pciutils-devel-2.2.3-4
libgtop2-2.14.4-2.fc6
audiofile-devel-0.2.6-5
gnome-speech-0.4.5-1.fc6
libgpg-error-devel-1.4-2
indent-2.2.9-14.fc6
flex-2.5.4a-41.fc6
pfmon-3.2-0.060621.7.1
sqlite-devel-3.3.6-2
ctags-5.6-1.1
fontconfig-devel-2.4.1-3.fc6
metacity-2.16.0-5.fc6
gcc-java-4.1.1-30
php-cli-5.1.6-3
system-config-language-1.1.11-2
postgresql-libs-8.1.4-1.1
php-5.1.6-3
docbook-utils-0.6.14-5.1
MySQL-python-1.2.1-1
system-config-printer-0.7.32-1
linuxdoc-tools-0.9.21-7.1
python-ldap-2.2.0-2.1
jlex-javadoc-1.2.6-5jpp.1
tomcat5-servlet-2.4-api-5.5.17-6jpp.2
libXrender-devel-0.9.1-3.1
kdnssd-avahi-0.1.3-0.1.20060713svn.fc6
libICE-devel-1.0.1-2.1
system-config-users-1.2.46-1.fc6
avahi-devel-0.6.11-6.fc6
libXrandr-devel-1.1.1-3.1
lucene-1.4.3-1jpp.14
esc-1.0.0-16.fc6
xorg-x11-fonts-75dpi-7.1-2
xalan-j2-2.7.0-6jpp.1
ant-apache-log4j-1.6.5-2jpp.2
ant-apache-regexp-1.6.5-2jpp.2
ant-javamail-1.6.5-2jpp.2
geronimo-specs-1.0-0.M2.2jpp.12
gnome-vfs2-smb-2.16.0-4.fc6
gnome-utils-2.16.0-1.fc6
openoffice.org-draw-2.0.4-5.3
yelp-2.16.0-4.fc6
gnome-terminal-2.16.0-2.fc6
krb5-auth-dialog-0.7-1
alacarte-0.10.0-1.fc6
rhythmbox-0.9.5-4.fc6
linuxwacom-0.7.4_1-2.1
atk-devel-1.12.2-1.fc6
boost-devel-1.33.1-6.1
libgtop2-devel-2.14.4-2.fc6
patchutils-0.2.31-2.2.2
libcap-devel-1.10-25
diffstat-1.41-1.2.2
mx4j-manual-3.0.1-6jpp.4
autoconf-2.59-12
libgcj-src-4.1.1-30
oprofile-gui-0.9.1-16
libuser-devel-0.54.7-2
junit-javadoc-3.8.2-3jpp.1
classpathx-mail-javadoc-1.1.1-4jpp.2
mx4j-javadoc-3.0.1-6jpp.4
xml-commons-resolver-javadoc-1.1-1jpp.12
startup-notification-devel-0.8-4.1
jakarta-commons-codec-1.3-7jpp.2
SDL-devel-1.2.10-6.2
rpm-build-4.4.2-32

libXtst-devel-1.0.1-3.1
dbus-devel-0.93-3.fc6
kdeaddons-3.5.4-1.fc6
xerces-j2-demo-2.7.1-7jpp.2
libcroco-devel-0.6.1-2.1
eclipse-cdt-3.1.1-1.fc6
kdevelop-3.3.4-1.fc6
kde-i18n-Polish-3.5.4-1
gpg-pubkey-4f2a6fd2-3f9d9d3b
lua-5.1.2-1.fc6
emerald-themes-0.2.0-1.fc6
libglademm24-2.6.3-2.fc6
bdock-0.2.0-1.fc6
libstatgrab-0.13-3.fc6
libsigsegv-2.4-2.fc6
python-twisted-mail-0.3.0-4.fc6
link-grammar-4.2.2-2.fc6
python-twisted-lore-0.2.0-4.fc6
brandy-1.0.19-4.fc6
binutils-2.17.50.0.6-2.fc6
abiword-2.4.6-1.fc6
buildbot-0.7.5-1.fc6
autogen-5.8.8-1.fc6
calcurse-1.8-2.fc6
ipw3945-firmware-1.14.2-1
wine-jack-0.9.36-2.fc6
R-2.5.0-3.fc6
iptraf-3.0.0-4.1
kismet-0.0.2006.04.R1-4.fc6
libmad-devel-0.15.1b-3.2.fc4.qyz
qt-3.3.8-2.fc6
arts-devel-1.5.7-0.1.fc6
nas-1.9-1.fc6
libutempter-devel-1.1.4-3.fc6
xine-0.99.5-1.fc6
xdg-utils-1.0.2-2.fc6
xmms-flac-1.1.2-27.fc6
directfb-0.9.25.1-3.fc6
xine-lib-extras-1.1.6-2.fc6
k3b-extras-0.12.17-1.fc6

Załącznik 2

Skrypty do programowania ARM na platformę Windows

Spis treści

Skrypty do programowania procesorów AT91SAM7S64-AT91SAM7S256

openocd_at91sam7s_dbg_ftdi.cfg

```
##openocd_at91sam7s_dbg_ftdi.cfg

#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface ft2232
ft2232_device_desc "Amontec JTAGkey A"
ft2232_layout jtagkey
ft2232_vid_pid 0x0403 0xcff8
jtag_speed 0
jtag_nsrst_delay 200
jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA:
# flash protect 0 0 1 off
```

For more information about the configuration files, take a look at:
http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger

Koniec openocd_at91sam7s_dbg_ftdi.cfg

openocd_at91sam7s_dbg_wiggler.cfg

```
##openocd_at91sam7s_dbg_wiggler.cfg
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger


## Koniec openocd_at91sam7s_dbg_wiggler.cfg
```

openocd_at91sam7s_ecr.script

```
## openocd_at91sam7s_ecr.script
# Init for debug from the openocd-sources, PLLR modified by mthomas
mww 0xfffffd44 0x00008000          # disable watchdog
mww 0xfffffd08 0xa5000001# enable user reset
mww 0xfffffc20 0x00000601# CKGR_MOR : enable the main oscillator
sleep 10
mww 0xfffffc2c 0x00481c0e          # CKGR_PLLR: 96.1097 MHz
sleep 10
mww 0xfffffc30 0x00000007# PMC_MCKR : MCK = PLL / 2 ~= 48 MHz
sleep 10
mww 0xffffff60 0x003c0100 # MC_FMR: flash mode (FWS=1,FMCN=60)
# arm7_9 force_hw_bkpts enable     # program resides in flash


## Koniec openocd_at91sam7s_ecr.script
```

openocd_at91sam7s_flash.script

```
##openocd_at91sam7s_flash.script
#
# The following command wills be executed on
# reset (because of run_and_init in the config-file)
# - halt target
# - init ecr
# - flash content of file main.bin into target-memory
# - shutdown openocd
#
# created by Martin Thomas
# http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects
# based on information from Dominic Rath
#

halt
sleep 10

# Init - taken form the script openocd_at91sam7_ecr.script
mww 0xfffffd44 0x00008000       # disable watchdog
mww 0xfffffd08 0xa5000001# enable user reset
mww 0xfffffc20 0x00000601# CKGR_MOR : enable the main oscillator
sleep 10
mww 0xfffffc2c 0x00481c0e       # CKGR_PLLR: 96.1097 MHz
sleep 10
mww 0xfffffc30 0x00000007# PMC_MCKR : MCK = PLL / 2 ~= 48 MHz
sleep 10
mww 0xffffff60 0x003c0100 # MC_FMR: flash mode (FWS=1,FMCN=60)
# arm7_9 force_hw_bkpts enable    # program resides in flash

# AT91SAM7 flash command-"batch"
# adapted by Martin Thomas based on information from Dominic Rath - Thanks
arm7_9 dcc_downloads enable
sleep 10
poll
flash probe 0
flash write 0 main.bin 0x0
reset run
sleep 10
shutdown


### openocd_at91sam7s_flash.script
```

openocd_at91sam7s_flash_ftdi.cfg

```
####openocd_at91sam7s_flash_ftdi.cfg

# Flash AT91SAM7S memory using openocd
# and a FTDI FT2232-based JTAG-interface
#
# created by Martin Thomas
# based on information from Dominic Rath
#
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface ft2232
ft2232_device_desc "Dual RS232"
ft2232_layout jtagkey
ft2232_vid_pid 0x0403 0xc6010
jtag_speed 0
jtag_nsrst_delay 200
jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_init 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
target_script 0 reset openocd_at91sam7s_flash.script
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA
```

```
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
### Koniec openocd_at91sam7s_flash_ftdi.cfg
```

openocd_at91sam7s_flash_wiggler.cfg

```
####openocd_at91sam7s_flash_wiggler.cfg
# Flash AT91SAM7S memory using openocd
# and a Wiggler-type JTAG-interface
#
# created by Martin Thomas
# based on information from Dominic Rath
#

#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_init 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
target_script 0 reset openocd_at91sam7s_flash.script
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
```

SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
set/clear NVM-Bits:
at91sam7 gpnvm <num> <bit> <set|clear>
disable locking from SAM-BA
flash protect 0 0 1 off

For more information about the configuration files, take a look at:
http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
###Koniec openocd_at91sam7s_flash_wiggler.cfg

Skrypty do programowania AT91SAM7SA3

openocd_at91sam7a3_dbg_ftdi.cfg

###openocd_at91sam7a3_dbg_ftdi.cfg

#daemon configuration

telnet_port 4444

gdb_port 3333

#interface

interface ft2232

ft2232_device_desc "Amontec JTAGkey A"

ft2232_layout jtagkey

ft2232_vid_pid 0x0403 0xcff8

jtag_speed 0

jtag_nsrst_delay 200

jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately

reset_config srst_only srst_pulls_trst

#jtag scan chain

#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)

jtag_device 4 0x1 0xf 0xe

#target configuration

daemon_startup reset

```
#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA:
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
### Koniec openocd_at91sam7a3_dbg_ftdi.cfg
```

openocd_at91sam7a3_dbg_wiggler.cfg

```
### openocd_at91sam7a3_dbg_wiggler.cfg
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
```

```
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger

### Koniec openocd_at91sam7a3_dbg_wiggler.cfg
```

openocd_at91sam7a3_ecr.script

```
### openocd_at91sam7a3_ecr.script
### # Init for debug from the openocd-sources, PLLR modified by mthomas
mww 0xfffffd44 0x00008000           # disable watchdog
mww 0xfffffd08 0xa5000001# enable user reset
mww 0xfffffc20 0x00000601# CKGR_MOR : enable the main oscillator
sleep 10
mww 0xfffffc2c 0x00481c0e           # CKGR_PLLR: 96.1097 MHz
sleep 10
mww 0xfffffc30 0x00000007# PMC_MCKR : MCK = PLL / 2 ~= 48 MHz
sleep 10
mww 0xffffff60 0x003c0100 # MC_FMR: flash mode (FWS=1,FMCN=60)
# arm7_9 force_hw_bkpts enable      # program resides in flash

### Koniec openocd_at91sam7a3_ecr.script
```

openocd_at91sam7a3_flash.script

```
### openocd_at91sam7a3_flash.script
#
```

```
# The following command wills be executed on
# reset (because of run_and_init in the config-file)
# - halt target
# - init ecr
# - flash content of file main.bin into target-memory
# - shutdown openocd
#
# created by Martin Thomas
# http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects
# based on information from Dominic Rath
#

halt
sleep 10

# Init - taken form the script openocd_at91sam7_ecr.script
mww 0xfffffd44 0x00008000           # disable watchdog
mww 0xfffffd08 0xa5000001# enable user reset
mww 0xfffffc20 0x00000601# CKGR_MOR : enable the main oscillator
sleep 10
mww 0xfffffc2c 0x00481c0e           # CKGR_PLLR: 96.1097 MHz
sleep 10
mww 0xfffffc30 0x00000007# PMC_MCKR : MCK = PLL / 2 ~= 48 MHz
sleep 10
mww 0xffffff60 0x003c0100 # MC_FMR: flash mode (FWS=1,FMCN=60)
# arm7_9 force_hw_bkpts enable      # program resides in flash

# AT91SAM7 flash command-"batch"
# adapted by Martin Thomas based on information from Dominic Rath - Thanks
arm7_9 dcc_downloads enable
sleep 10
poll
flash probe 0
flash write 0 main.bin 0x0
reset run
sleep 10
shutdown

#### Koniec openocd_at91sam7a3_flash.script
```

openocd_at91sam7a3_flash_ftdi.cfg

```
##### openocd_at91sam7a3_flash_ftdi.cfg
# Flash AT91SAM7S memory using openocd
# and a FTDI FT2232-based JTAG-interface
#
# created by Martin Thomas
# based on information from Dominic Rath
#
```

#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface ft2232
ft2232_device_desc "Amontec JTAGkey A"
ft2232_layout jtagkey
ft2232_vid_pid 0x0403 0xcff8
jtag_speed 0
jtag_nsrst_delay 200
jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_init 0 arm7tdmi
run_and_halt_time 0 30

flash-options AT91
target_script 0 reset openocd_at91sam7a3_flash.script
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

Information:
erase command (telnet-interface) for complete flash:
flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
set/clear NVM-Bits:
at91sam7 gpnvm <num> <bit> <set|clear>
disable locking from SAM-BA
flash protect 0 0 1 off

For more information about the configuration files, take a look at:
http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger

Koniec openocd_at91sam7a3_flash_ftdi.cfg

openocd_at91sam7a3_flash_wiggler.cfg

```
#### openocd_at91sam7a3_flash_wiggler.cfg
# Flash AT91SAM7S memory using openocd
# and a Wiggler-type JTAG-interface
#
# created by Martin Thomas
# based on information from Dominic Rath
#

#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_init 0 arm7tdmi
run_and_halt_time 0 30

# flash-options AT91
target_script 0 reset openocd_at91sam7a3_flash.script
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
```

Koniec openocd_at91sam7a3_flash_wiggler.cfg

AT91SAM7A3-RAM.ld

```
### AT91SAM7A3-RAM.ld
/*---------------------------------------------------------------------------*/
/*-         ATMEL Microcontroller Software Support  -  ROUSSET  - */
/*---------------------------------------------------------------------------*/
/* The software is delivered "AS IS" without warranty or condition of any    */
/* kind, either express, implied or statutory. This includes without */
/* limitation any warranty or condition with respect to merchantability or   */
/* fitness for any particular purpose, or against the infringements of */
/* intellectual property rights of others.                                  */
/*---------------------------------------------------------------------------*/
/*- File source          : GCC_FLASH.ld */
/*- Object               : Linker Script File for Flash Workspace */
/*- Compilation flag     : None */
/*- */
/*- 1.0 11/Mar/05 JPP    : Creation SAM7A3 */
/*---------------------------------------------------------------------------*/
/* Additional modification by Martin Thomas

/* Memory Definitions */

MEMORY
{
  ROM (rx) : ORIGIN = 0x00000000, LENGTH = 0x00007000
  RAM (rw) : ORIGIN = 0x00007000, LENGTH = 0x00001000
  STACK (rw) : ORIGIN = 0x00007000,LENGTH = 0x00000000
}


/* Section Definitions */

SECTIONS
{
  /* first section is .text which is used for code */
   .text :
  {
        *SAM7A3Assembly.o (.text)          /* Startup code */
        *(.text .text.*)          /* remaining code */
        *(.gnu.linkonce.t.*)
        *(.glue_7)
        *(.glue_7t)
        *(.gcc_except_table)
        *(.rodata)                /* read-only data (constants) */
        *(.rodata*)
        *(.gnu.linkonce.r.*)
  } > RAM
```

```
. = ALIGN(4);

        /* .ctors .dtors are used for c++ constructors/destructors */
        /* added by Martin Thomas 4/2005 based on Anglia Design example */
        .ctors :
        {
                PROVIDE(__ctors_start__ = .);
                KEEP(*(SORT(.ctors.*)))
                KEEP(*(.ctors))
                PROVIDE(__ctors_end__ = .);
        } >RAM

        .dtors :
        {
                PROVIDE(__dtors_start__ = .);
                KEEP(*(SORT(.dtors.*)))
                KEEP(*(.dtors))
                PROVIDE(__dtors_end__ = .);
        } >RAM

  . = ALIGN(4);
        /* mthomas - end */

_etext = . ;
PROVIDE (etext = .);

/* .data section which is used for initialized data */
.data :
{
  _data = .;
  *(.data)
        *(.data.*)
        *(.gnu.linkonce.d*)
        SORT(CONSTRUCTORS) /* mt 4/2005 */
} > RAM

. = ALIGN(4);
_edata = . ;
PROVIDE (edata = .);

/* .bss section which is used for uninitialized data */
.bss (NOLOAD) :
{
  __bss_start = . ;
  __bss_start__ = . ;
  *(.bss)
        *(.gnu.linkonce.b*)
  *(COMMON)
  . = ALIGN(4);
```

```
} > RAM

. = ALIGN(4);
__bss_end__ = . ;
PROVIDE (__bss_end = .);

_end = . ;
PROVIDE (end = .);

. = ALIGN(4);
 .int_data :
 {
 *(.internal_ram_top)
 }> STACK

/* Stabs debugging sections.  */
.stab          0 : { *(.stab) }
.stabstr       0 : { *(.stabstr) }
.stab.excl     0 : { *(.stab.excl) }
.stab.exclstr  0 : { *(.stab.exclstr) }
.stab.index    0 : { *(.stab.index) }
.stab.indexstr 0 : { *(.stab.indexstr) }
.comment       0 : { *(.comment) }
/* DWARF debug sections.
   Symbols in the DWARF debugging sections are relative to the
beginning
   of the section so we begin them at 0.  */
/* DWARF 1 */
.debug          0 : { *(.debug) }
.line           0 : { *(.line) }
/* GNU DWARF 1 extensions */
.debug_srcinfo  0 : { *(.debug_srcinfo) }
.debug_sfnames  0 : { *(.debug_sfnames) }
/* DWARF 1.1 and DWARF 2 */
.debug_aranges  0 : { *(.debug_aranges) }
.debug_pubnames 0 : { *(.debug_pubnames) }
/* DWARF 2 */
.debug_info     0 : { *(.debug_info .gnu.linkonce.wi.*) }
.debug_abbrev   0 : { *(.debug_abbrev) }
.debug_line     0 : { *(.debug_line) }
.debug_frame    0 : { *(.debug_frame) }
.debug_str      0 : { *(.debug_str) }
.debug_loc      0 : { *(.debug_loc) }
.debug_macinfo  0 : { *(.debug_macinfo) }
/* SGI/MIPS DWARF 2 extensions */
.debug_weaknames 0 : { *(.debug_weaknames) }
.debug_funcnames 0 : { *(.debug_funcnames) }
.debug_typenames 0 : { *(.debug_typenames) }
.debug_varnames  0 : { *(.debug_varnames) }
```

```
}
## Koniec AT91SAM7A3-RAM.ld
```

AT91SAM7A3-ROM.ld

```
## AT91SAM7A3-ROM.ld

## /*------------------------------------------------------------------------*/
/*-         ATMEL Microcontroller Software Support  -  ROUSSET  - */
/*------------------------------------------------------------------------*/
/* The software is delivered "AS IS" without warranty or condition of any    */
/* kind, either express, implied or statutory. This includes without */
/* limitation any warranty or condition with respect to merchantability or   */
/* fitness for any particular purpose, or against the infringements of */
/* intellectual property rights of others.                               */
/*------------------------------------------------------------------------*/
/*- File source          : GCC_FLASH.ld */
/*- Object               : Linker Script File for Flash Workspace */
/*- Compilation flag     : None */
/*- */
/*- 1.0 11/Mar/05 JPP    : Creation SAM7A3 */
/*------------------------------------------------------------------------*/
/* Additional modification by Martin Thomas

/* Memory Definitions */

MEMORY
{
  ROM (rx) : ORIGIN = 0x00000000, LENGTH = 0x00040000
  RAM (rw) : ORIGIN = 0x00200000, LENGTH = 0x00008000
  STACK (rw) : ORIGIN = 0x00208000,LENGTH = 0x00000000
}


/* Section Definitions */

SECTIONS
{
  /* first section is .text which is used for code */
   .text :
  {
         *SAM7A3Assembly.o (.text)           /* Startup code */
         *(.text .text.*)          /* remaining code */
         *(.gnu.linkonce.t.*)
         *(.glue_7)
         *(.glue_7t)
         *(.gcc_except_table)
         *(.rodata)                /* read-only data (constants) */
         *(.rodata*)
```

```
        *(.gnu.linkonce.r.*)
} > ROM

. = ALIGN(4);

        /* .ctors .dtors are used for c++ constructors/destructors */
        /* added by Martin Thomas 4/2005 based on Anglia Design example */
        .ctors :
        {
                PROVIDE(__ctors_start__ = .);
                KEEP(*(SORT(.ctors.*)))
                KEEP(*(.ctors))
                PROVIDE(__ctors_end__ = .);
        } >ROM

        .dtors :
        {
                PROVIDE(__dtors_start__ = .);
                KEEP(*(SORT(.dtors.*)))
                KEEP(*(.dtors))
                PROVIDE(__dtors_end__ = .);
        } >ROM

  . = ALIGN(4);
        /* mthomas - end */

_etext = . ;
PROVIDE (etext = .);

/* .data section which is used for initialized data */
.data : AT (_etext)
{
  _data = .;
  *(.data)
        *(.data.*)
        *(.gnu.linkonce.d*)
        SORT(CONSTRUCTORS) /* mt 4/2005 */
        . = ALIGN(4);

} > RAM

. = ALIGN(4);
_edata = . ;
PROVIDE (edata = .);

/* .bss section which is used for uninitialized data */
.bss (NOLOAD) :
{
  __bss_start = . ;
  __bss_start__ = . ;
```

```
   *(.bss)
          *(.gnu.linkonce.b*)
   *(COMMON)
   . = ALIGN(4);
  } > RAM

  . = ALIGN(4);
  __bss_end__ = . ;
  PROVIDE (__bss_end = .);

  _end = . ;
  PROVIDE (end = .);

  . = ALIGN(4);
   .int_data :
   {
   *(.internal_ram_top)
   }> STACK

  /* Stabs debugging sections.  */
  .stab          0 : { *(.stab) }
  .stabstr       0 : { *(.stabstr) }
  .stab.excl     0 : { *(.stab.excl) }
  .stab.exclstr  0 : { *(.stab.exclstr) }
  .stab.index    0 : { *(.stab.index) }
  .stab.indexstr 0 : { *(.stab.indexstr) }
  .comment       0 : { *(.comment) }
  /* DWARF debug sections.
     Symbols in the DWARF debugging sections are relative to the
beginning
     of the section so we begin them at 0.  */
  /* DWARF 1 */
  .debug          0 : { *(.debug) }
  .line           0 : { *(.line) }
  /* GNU DWARF 1 extensions */
  .debug_srcinfo  0 : { *(.debug_srcinfo) }
  .debug_sfnames  0 : { *(.debug_sfnames) }
  /* DWARF 1.1 and DWARF 2 */
  .debug_aranges  0 : { *(.debug_aranges) }
  .debug_pubnames 0 : { *(.debug_pubnames) }
  /* DWARF 2 */
  .debug_info     0 : { *(.debug_info .gnu.linkonce.wi.*) }
  .debug_abbrev   0 : { *(.debug_abbrev) }
  .debug_line     0 : { *(.debug_line) }
  .debug_frame    0 : { *(.debug_frame) }
  .debug_str      0 : { *(.debug_str) }
  .debug_loc      0 : { *(.debug_loc) }
  .debug_macinfo  0 : { *(.debug_macinfo) }
  /* SGI/MIPS DWARF 2 extensions */
  .debug_weaknames 0 : { *(.debug_weaknames) }
```

```
  .debug_funcnames 0 : { *(.debug_funcnames) }
  .debug_typenames 0 : { *(.debug_typenames) }
  .debug_varnames  0 : { *(.debug_varnames) }

}
### Koniec AT91SAM7A3-ROM.ld
```

Skrypty dla LPC2138

openocd_lpc2138_dbg_ftdi.cfg

```
###openocd_lpc2138_dbg_ftdi.cfg
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface ft2232
ft2232_device_desc "Amontec JTAGkey A"
ft2232_layout jtagkey
ft2232_vid_pid 0x0403 0xcff8
jtag_speed 0
jtag_nsrst_delay 200
jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi
```

```
run_and_halt_time 0 30

# flash-options AT91
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA:
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
### Koniec openocd_lpc2138_dbg_ftdi.cfg
```

openocd_lpc2138_dbg_wiggler.cfg

```
###openocd_lpc2138_dbg_wiggler.cfg
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst srst_pulls_trst
```

```
#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi-s_r4
run_and_halt_time 0 30


# flash-options LPC2138
working_area 0 0x40000000 0x4000 nobackup
# LPC2138 @ 12MHz / 0x7D000 from 500*1024 (not 512!)
flash bank lpc2000 0x0 0x7D000 0 0 lpc2000_v2 0 12000 calc_checksum

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
##Koniec openocd_lpc2138_dbg_wiggler.cfg
```

openocd_lpc2138_flash.script

```
###openocd_lpc2138_flash.script
#
# The following command wills be executed on
# reset (because of run_and_init in the config-file)
# - wait for target halt
# - erase memory
# - flash content of file main.bin into target-memory
# - shutdown openocd
#
# created by Martin Thomas
# http://www.siwawi.arubi.uni-kl.de/avr_projects/arm_projects
```

```
# based on information from Dominic Rath
#

arm7_9 dcc_downloads enable
wait_halt
sleep 10
poll
flash probe 0
flash erase 0 0 0
flash write 0 main.bin 0x0
reset run
sleep 10
shutdown
### Koniec openocd_lpc2138_flash.script
```

openocd_lpc2138_flash_ftdi.cfg

```
## openocd_lpc2138_flash_ftdi.cfg
#
# Flash LPC2138 memory using openocd
# and a FTDI FT2232-based JTAG-interface
#
# created by Martin Thomas
# based on information from Dominic Rath
#

#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface ft2232
#ft2232_device_desc "Amontec JTAGkey A"
ft2232_device_desc "Dual RS232 A"
ft2232_layout jtagkey
ft2232_vid_pid 0x0403 0x6010
```

```
jtag_speed 3
jtag_nsrst_delay 200
jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_init 0 arm7tdmi-s_r4
run_and_halt_time 0 30

# flash-options LPC2138
target_script 0 reset openocd_lpc2138_flash.script
working_area 0 0x40000000 0x4000 nobackup
# LPC2138 @ 12MHz / 0x7D000 from 500*1024 (not 512!)
flash bank lpc2000 0x0 0x7D000 0 0 lpc2000_v2 0 12000 calc_checksum

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
## Koniec openocd_lpc2138_flash_ftdi.cfg
```

openocd_lpc2138_flash_wiggler.cfg

```
### openocd_lpc2138_flash_wiggler.cfg
#
# Flash LPC2138 memory using openocd
# and a Wiggler-type JTAG-interface
#
```

```
# created by Martin Thomas
# based on information from Dominic Rath
#

#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_init 0 arm7tdmi-s_r4
run_and_halt_time 0 30

# flash-options LPC2138
target_script 0 reset openocd_lpc2138_flash.script
working_area 0 0x40000000 0x4000 nobackup
# LPC2138 @ 12MHz / 0x7D000 from 500*1024 (not 512!)
flash bank lpc2000 0x0 0x7D000 0 0 lpc2000_v2 0 12000 calc_checksum
```

```
# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
## Koniec openocd_lpc2138_flash_wiggler.cfg
```

Skrypty dla STR912

openocd_str912_dbg_ftdi.cfg

```
### openocd_str912_dbg_ftdi.cfg
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface ft2232
ft2232_device_desc "Amontec JTAGkey A"
ft2232_layout jtagkey
ft2232_vid_pid 0x0403 0xcff8
jtag_speed 0
jtag_nsrst_delay 200
jtag_ntrst_delay 200

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config srst_only srst_pulls_trst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 4 0x1 0xf 0xe

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
target arm7tdmi little run_and_halt 0 arm7tdmi
run_and_halt_time 0 30
```

```
# flash-options AT91
working_area 0 0x00200000 0x4000 nobackup
flash bank at91sam7 0 0 0 0 0

# Information:
# erase command (telnet-interface) for complete flash:
# flash erase <num> 0 numlockbits-1 (can be seen from output of flash info 0)
# SAM7S64 with 16 lockbits and bank 0: flash erase 0 0 15
# set/clear NVM-Bits:
# at91sam7 gpnvm <num> <bit> <set|clear>
# disable locking from SAM-BA:
# flash protect 0 0 1 off

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger

### Koniec  openocd_str912_dbg_ftdi.cfg
```

openocd_str912_dbg_wiggler.cfg

```
### openocd_str912_dbg_wiggler.cfg
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst
```

```
#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 8 0x1 0x1 0xfe
jtag_device 4 0x1 0xf 0xe
jtag_device 5 0x1 0x1 0x1e
#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm966e <endianness> <reset mode> <chainpos> <variant>
target arm966e little reset_halt 1 arm966e
run_and_halt_time 0 30

working_area 0 0x50000000 16384 nobackup

#flash bank <driver> <base> <size> <chip_width> <bus_width>
flash bank str9x 0x00000000 0x00080000 0 0 0

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger


### openocd_str912_dbg_wiggler.cfg
```

openocd_str912_flash.script

```
### openocd_str912_flash.script
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0
```

```
#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 8 0x1 0x1 0xfe
jtag_device 4 0x1 0xf 0xe
jtag_device 5 0x1 0x1 0x1e
#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm966e <endianness> <reset mode> <chainpos> <variant>
target arm966e little reset_halt 1 arm966e
run_and_halt_time 0 30

working_area 0 0x50000000 16384 nobackup

#flash bank <driver> <base> <size> <chip_width> <bus_width>
flash bank str9x 0x00000000 0x00080000 0 0 0

# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger

### Koniec openocd_str912_flash.script
```

openocd_str912_flash_ftdi.cfg

```
### openocd_str912_flash_ftdi.cfg
#
#debug 3
#daemon configuration
telnet_port 4444
gdb_port 3333
```

```
#interface
interface ft2232
ft2232_device_desc "Dual RS232 A"
#ft2232_layout jtagkey
ft2232_layout usbjtag

ft2232_vid_pid 0x0403 0x6010
jtag_speed 3


#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst
#reset_config trst_and_srst srst_pulls_trst
#reset_config srst_only

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)
jtag_device 8 0x1 0x01 0xfe
jtag_device 4 0x1 0x0f 0x0e
jtag_device 5 0x1 0x01 0x1e

#target configuration
daemon_startup reset

#target <type> <startup mode>
#target arm7tdmi <reset mode> <chainpos> <endianness> <variant>
#target arm966e little reset_halt 1 arm966e
target arm966e little reset_init 1 arm966e
run_and_halt_time 0 30

# working_area <target#> <address> <size> <'backup'|'nobackup'>
target_script 0 reset OpenOCD\openocd_str912_flash.script
working_area 0 0x50000000 16384 nobackup

#flash bank <driver> <base> <size> <chip_width> <bus_width>
```

flash bank str9x 0x00000000 0x00080000 0 0 0

For more information about the configuration files, take a look at:
http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
http://www.openhardware.net/Embedded_ARM/OpenOCD_JTAG/

Koniec openocd_str912_flash_ftdi.cfg

openocd_str912_flash_wiggler.cfg

```
### openocd_str912_flash_wiggler.cfg
#debug 3
#daemon configuration
telnet_port 4444
gdb_port 3333

#interface
interface parport
parport_port 0x378
parport_cable wiggler
jtag_speed 0

#use combined on interfaces or targets that can't set TRST/SRST separately
reset_config trst_and_srst

#jtag scan chain
#format L IRC IRCM IDCODE (Length, IR Capture, IR Capture Mask, IDCODE)

jtag_device 8 0x1 0x01 0xfe
jtag_device 4 0x1 0x0f 0x0e
jtag_device 5 0x1 0x01 0x1e

#target configuration
```

```
daemon_startup reset

#target <type> <startup mode>
#target arm966e <endianness> <reset mode> <chainpos> <variant>
#target arm966e little reset_halt 1 arm966e
target arm966e little reset_init 1 arm966e
run_and_halt_time 0 30

# working_area <target#> <address> <size> <'backup'|'nobackup'>
target_script 0 reset OpenOCD\openocd_str912_flash.script
working_area 0 0x50000000 16384 nobackup

#flash bank <driver> <base> <size> <chip_width> <bus_width>
flash bank str9x 0x00000000 0x00080000 0 0 0


# For more information about the configuration files, take a look at:
# http://openfacts.berlios.de/index-en.phtml?title=Open+On-Chip+Debugger
# http://www.openhardware.net/Embedded_ARM/OpenOCD_JTAG/


### Koniec openocd_str912_flash_wiggler.cfg
```

STR912-RAM.ld

```
### STR912-RAM.ld
/*** Linker Script File ***/
/* Hitex/Gn/07.09.2006 */
/* Memory Definitions */
/* STR912 */


/*******************************************
  Define Files
*******************************************/
```

```
/*****************************************
  Memory Definitions
*****************************************/
/* RAM usage */
MEMORY
{
 IntCodeRAM (rx) : ORIGIN = 0x40000000, LENGTH = 0x00003000
 IntDataRAM (rw) : ORIGIN = 0x40003000, LENGTH = 0x00001000
}


/*****************************************
  Section Definitions
*****************************************/

SECTIONS
{
/*****************************************/

.text :
   {

         __code_start__ = .;

         startup_str912.o       (.text)

             *.o      (.text)

         . = ALIGN(4);
         __code_end__ = .;

         *(.glue_7t) *(.glue_7)
```

```
} >IntCodeRAM =0

. = ALIGN(4);

/* .rodata section which is used for read-only data (constants) */

.rodata . :
{
    *(.rodata)
} >IntCodeRAM

. = ALIGN(4);

_etext = . ;
PROVIDE (etext = .);


  /*******************************************/
.data : AT (_etext)
{
  /* used for initialized data */
  __data_start__ = . ;
  PROVIDE (__data_start__ = .) ;
  *(.data)
  SORT(CONSTRUCTORS)
  __data_end__ = . ;
  PROVIDE (__data_end__ = .) ;
} >IntDataRAM
. = ALIGN(4);

_edata = . ;
 PROVIDE (edata = .);

  /*******************************************/
.bss :
```

```
  {
   /* used for uninitialized data */

   __bss_start = . ;
   __bss_start__ = . ;
   *(.bss)
   . = ALIGN(4);
   __bss_end__ = . ;

  } >IntDataRAM

  .bss2 :
  {
   /* used for uninitialized data */

   __bss2_start = . ;
   __bss2_start__ = . ;
   *(COMMON)
   . = ALIGN(4);
   __bss2_end__ = . ;

  } >IntDataRAM

/*******************************************/
  _end = .;
  PROVIDE (end = .);

/*******************************************/
  .comment       0 : { *(.comment) }
  /* DWARF debug sections.
     Symbols in the DWARF debugging sections are relative to the beginning
     of the section so we begin them at 0.  */
  /* DWARF 1 */
  .debug          0 : { *(.debug) }
  .line           0 : { *(.line) }
```

```
  /* GNU DWARF 1 extensions */
  .debug_srcinfo  0 : { *(.debug_srcinfo) }
  .debug_sfnames  0 : { *(.debug_sfnames) }
  /* DWARF 1.1 and DWARF 2 */
  .debug_aranges  0 : { *(.debug_aranges) }
  .debug_pubnames 0 : { *(.debug_pubnames) }
  /* DWARF 2 */
  .debug_info     0 : { *(.debug_info .gnu.linkonce.wi.*) }
  .debug_abbrev   0 : { *(.debug_abbrev) }
  .debug_line     0 : { *(.debug_line) }
  .debug_frame    0 : { *(.debug_frame) }
  .debug_str      0 : { *(.debug_str) }
  .debug_loc      0 : { *(.debug_loc) }
  .debug_macinfo  0 : { *(.debug_macinfo) }
}
### Koniec STR912-RAM.ld
```

STR912-ROM.ld

```
### STR912-ROM.ld
/*** Linker Script File ***/
/* Hitex/Gn/07.09.2006 */
/* Memory Definitions */
/* STR912 */

/* memory layout:
//
//   Exception vectors [0x000000--0x00001F]  RAM or ROM
//   ROMSTART--ROMEND  [0x008000--0x0FFFFF]  ROM (or other non-volatile memory)
//   RAMSTART--RAMEND  [0x100000--0x7FFFFF]  RAM (or other read/write memory)  */


/******************************************
  Define Files
******************************************/
```

```
/*****************************************
  Memory Definitions
*****************************************/

/* Flash usage */
MEMORY
{
 IntCodeFlash (rx) : ORIGIN = 0x00000000, LENGTH = 512k
 IntDataRAM (rw)   : ORIGIN = 0x40000000, LENGTH = 96k
 IntDataEth (!rx)  : ORIGIN = 0x7C000000, LENGTH = 0x42F   /* AHB nonbuffered Ethernet
RAM */
}

/* this address is used in startup for initilizing stack */
/* stack is at the end of data range                                  */

PROVIDE(_top_stack_ = 0x40018000 -4);

SECTIONS
{

 /* first section is .text which is used for code */
 .start : { *(.startup)} >IntCodeFlash = 0
 .text :
 {
/* here is the path to change and Processor-specific ISR_XXX-file */
      startup_str912.o (.text) /* Startup code */
   *(.text)          /* remaining code */
   *(.glue_7t) *(.glue_7)

 } >IntCodeFlash =0

      __end_of_text__ = .;
```

```
. = ALIGN(4);

/* .rodata section which is used for read-only data (constants) */

.rodata . :
{
  *(.rodata)
} >IntCodeFlash

. = ALIGN(4);

_etext = . ;
PROVIDE (etext = .);

/* .data section which is used for initialized data */

.data : AT (_etext)
{
  _data = . ;
  __data_beg_src__ = __end_of_text__;
  __data_start__ = . ;
  PROVIDE (__data_start__ = .) ;
  *(.data)
  SORT(CONSTRUCTORS)
} >IntDataRAM
. = ALIGN(4);

_edata = . ;
 PROVIDE (edata = .);

/* .bss section which is used for uninitialized data */

.bss :
{
  __bss_start = . ;
```

```
    __bss_start__ = . ;
    *(.bss)
    *(COMMON)
  } >IntDataRAM
  . = ALIGN(4);
  __bss_end__ = . ;
  __bss_end = . ;

  _end = .;
  PROVIDE (end = .);

.bss2 :
  {
    /* used for uninitialized data */

    __bss2_start = . ;
    __bss2_start__ = . ;
    *(COMMON)
    . = ALIGN(4);
    __bss2_end__ = . ;

  } >IntDataRAM

  /* Stabs debugging sections.  */
  .stab          0 : { *(.stab) }
  .stabstr       0 : { *(.stabstr) }
  .stab.excl     0 : { *(.stab.excl) }
  .stab.exclstr  0 : { *(.stab.exclstr) }
  .stab.index    0 : { *(.stab.index) }
  .stab.indexstr 0 : { *(.stab.indexstr) }
  .comment       0 : { *(.comment) }
  /* DWARF debug sections.
     Symbols in the DWARF debugging sections are relative to the beginning
     of the section so we begin them at 0.  */
  /* DWARF 1 */
```

```
  .debug          0 : { *(.debug) }
  .line           0 : { *(.line) }
  /* GNU DWARF 1 extensions */
  .debug_srcinfo  0 : { *(.debug_srcinfo) }
  .debug_sfnames  0 : { *(.debug_sfnames) }
  /* DWARF 1.1 and DWARF 2 */
  .debug_aranges  0 : { *(.debug_aranges) }
  .debug_pubnames 0 : { *(.debug_pubnames) }
  /* DWARF 2 */
  .debug_info     0 : { *(.debug_info .gnu.linkonce.wi.*) }
  .debug_abbrev   0 : { *(.debug_abbrev) }
  .debug_line     0 : { *(.debug_line) }
  .debug_frame    0 : { *(.debug_frame) }
  .debug_str      0 : { *(.debug_str) }
  .debug_loc      0 : { *(.debug_loc) }
  .debug_macinfo  0 : { *(.debug_macinfo) }
  /* SGI/MIPS DWARF 2 extensions */
  .debug_weaknames 0 : { *(.debug_weaknames) }
  .debug_funcnames 0 : { *(.debug_funcnames) }
  .debug_typenames 0 : { *(.debug_typenames) }
  .debug_varnames  0 : { *(.debug_varnames) }
}
```

Koniec STR912-ROM.ld

Załącznik 3

semtest.c

```
/*
	FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

	This file is part of the FreeRTOS.org distribution.

	FreeRTOS.org is free software; you can redistribute it and/or modify
	it under the terms of the GNU General Public License as published by
	the Free Software Foundation; either version 2 of the License, or
	(at your option) any later version.

	FreeRTOS.org is distributed in the hope that it will be useful,
	but WITHOUT ANY WARRANTY; without even the implied warranty of
	MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
	GNU General Public License for more details.

	You should have received a copy of the GNU General Public License
	along with FreeRTOS.org; if not, write to the Free Software
	Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA

	A special exception to the GPL can be applied should you wish to distribute
	a combined work that includes FreeRTOS.org, without being obliged to provide
	the source code for any proprietary components.  See the licensing section
	of http://www.FreeRTOS.org for full details of how and when the exception
	can be applied.

	***************************************************************************
	See http://www.FreeRTOS.org for documentation, latest information, license
	and contact details.  Please ensure to read the configuration and relevant
	port sections of the online documentation.
	***************************************************************************
*/

/*
 * Creates two sets of two tasks.  The tasks within a set share a variable, access
 * to which is guarded by a semaphore.
 *
 * Each task starts by attempting to obtain the semaphore.  On obtaining a
 * semaphore a task checks to ensure that the guarded variable has an expected
 * value.  It then clears the variable to zero before counting it back up to the
 * expected value in increments of 1.  After each increment the variable is checked
 * to ensure it contains the value to which it was just set. When the starting
```

```
 * value is again reached the task releases the semaphore giving the other task in
 * the set a chance to do exactly the same thing.  The starting value is high
 * enough to ensure that a tick is likely to occur during the incrementing loop.
 *
 * An error is flagged if at any time during the process a shared variable is
 * found to have a value other than that expected.  Such an occurrence would
 * suggest an error in the mutual exclusion mechanism by which access to the
 * variable is restricted.
 *
 * The first set of two tasks poll their semaphore.  The second set use blocking
 * calls.
 *
 */


#include <stdlib.h>

/* Scheduler include files. */
#include "FreeRTOS.h"
#include "task.h"
#include "semphr.h"

/* Demo app include files. */
#include "semtest.h"

/* The value to which the shared variables are counted. */
#define semtstBLOCKING_EXPECTED_VALUE               ( ( unsigned portLONG ) 0xfff )
#define semtstNON_BLOCKING_EXPECTED_VALUE    ( ( unsigned portLONG ) 0xff  )

#define semtstSTACK_SIZE                     configMINIMAL_STACK_SIZE

#define semtstNUM_TASKS                              ( 4 )

#define semtstDELAY_FACTOR                   ( ( portTickType ) 10 )

/* The task function as described at the top of the file. */
static portTASK_FUNCTION_PROTO( prvSemaphoreTest, pvParameters );

/* Structure used to pass parameters to each task. */
typedef struct SEMAPHORE_PARAMETERS
{
       xSemaphoreHandle xSemaphore;
       volatile unsigned portLONG *pulSharedVariable;
       portTickType xBlockTime;
} xSemaphoreParameters;
```

```
/* Variables used to check that all the tasks are still running without errors. */
static volatile portSHORT sCheckVariables[ semtstNUM_TASKS ] = { 0 };
static volatile portSHORT sNextCheckVariable = 0;

/*-----------------------------------------------------------*/

void vStartSemaphoreTasks( unsigned portBASE_TYPE uxPriority )
{
xSemaphoreParameters *pxFirstSemaphoreParameters, *pxSecondSemaphoreParameters;
const portTickType xBlockTime = ( portTickType ) 100;

	/* Create the structure used to pass parameters to the first two tasks. */
	pxFirstSemaphoreParameters = ( xSemaphoreParameters * )
pvPortMalloc( sizeof( xSemaphoreParameters ) );

	if( pxFirstSemaphoreParameters != NULL )
	{
	 /* Create the semaphore used by the first two tasks. */
	 vSemaphoreCreateBinary( pxFirstSemaphoreParameters->xSemaphore );

	 if( pxFirstSemaphoreParameters->xSemaphore != NULL )
	 {
		/* Create the variable which is to be shared by the first two tasks. */
		pxFirstSemaphoreParameters->pulSharedVariable = ( unsigned portLONG * ) pvPortMalloc(
sizeof( unsigned portLONG ) );

		/* Initialise the share variable to the value the tasks expect. */
		*( pxFirstSemaphoreParameters->pulSharedVariable ) =
semtstNON_BLOCKING_EXPECTED_VALUE;

		/* The first two tasks do not block on semaphore calls. */
		pxFirstSemaphoreParameters->xBlockTime = ( portTickType ) 0;

		/* Spawn the first two tasks.  As they poll they operate at the idle priority. */
		xTaskCreate( prvSemaphoreTest, ( signed portCHAR * ) "PolSEM1", semtstSTACK_SIZE,
( void * ) pxFirstSemaphoreParameters, tskIDLE_PRIORITY, ( xTaskHandle * ) NULL );
		xTaskCreate( prvSemaphoreTest, ( signed portCHAR * ) "PolSEM2", semtstSTACK_SIZE,
( void * ) pxFirstSemaphoreParameters, tskIDLE_PRIORITY, ( xTaskHandle * ) NULL );
	 }
	}

	/* Do exactly the same to create the second set of tasks, only this time
	provide a block time for the semaphore calls. */
	pxSecondSemaphoreParameters = ( xSemaphoreParameters * )
pvPortMalloc( sizeof( xSemaphoreParameters ) );
	if( pxSecondSemaphoreParameters != NULL )
	{
	 vSemaphoreCreateBinary( pxSecondSemaphoreParameters->xSemaphore );
```

```
		if( pxSecondSemaphoreParameters->xSemaphore != NULL )
		{
			pxSecondSemaphoreParameters->pulSharedVariable = ( unsigned portLONG * )
pvPortMalloc( sizeof( unsigned portLONG ) );
			*( pxSecondSemaphoreParameters->pulSharedVariable ) = semtstBLOCKING_EXPECTED_VALUE;
			pxSecondSemaphoreParameters->xBlockTime = xBlockTime / portTICK_RATE_MS;

			xTaskCreate( prvSemaphoreTest, ( signed portCHAR * ) "BlkSEM1", semtstSTACK_SIZE,
( void * ) pxSecondSemaphoreParameters, uxPriority, ( xTaskHandle * ) NULL );
			xTaskCreate( prvSemaphoreTest, ( signed portCHAR * ) "BlkSEM2", semtstSTACK_SIZE,
( void * ) pxSecondSemaphoreParameters, uxPriority, ( xTaskHandle * ) NULL );
		}
	}
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( prvSemaphoreTest, pvParameters )
{
xSemaphoreParameters *pxParameters;
volatile unsigned portLONG *pulSharedVariable, ulExpectedValue;
unsigned portLONG ulCounter;
portSHORT sError = pdFALSE, sCheckVariableToUse;

	/* See which check variable to use.  sNextCheckVariable is not semaphore
	protected! */
	portENTER_CRITICAL();
		sCheckVariableToUse = sNextCheckVariable;
		sNextCheckVariable++;
	portEXIT_CRITICAL();

	/* A structure is passed in as the parameter.  This contains the shared
	variable being guarded. */
	pxParameters = ( xSemaphoreParameters * ) pvParameters;
	pulSharedVariable = pxParameters->pulSharedVariable;

	/* If we are blocking we use a much higher count to ensure loads of context
	switches occur during the count. */
	if( pxParameters->xBlockTime > ( portTickType ) 0 )
	{
		ulExpectedValue = semtstBLOCKING_EXPECTED_VALUE;
	}
	else
	{
		ulExpectedValue = semtstNON_BLOCKING_EXPECTED_VALUE;
	}

	for( ;; )
	{
```

```
/* Try to obtain the semaphore. */
if( xSemaphoreTake( pxParameters->xSemaphore, pxParameters->xBlockTime ) == pdPASS )
{
	/* We have the semaphore and so expect any other tasks using the
	shared variable to have left it in the state we expect to find
	it. */
	if( *pulSharedVariable != ulExpectedValue )
	{
		sError = pdTRUE;
	}

	/* Clear the variable, then count it back up to the expected value
	before releasing the semaphore.  Would expect a context switch or
	two during this time. */
	for( ulCounter = ( unsigned portLONG ) 0; ulCounter <= ulExpectedValue; ulCounter++ )
	{
		*pulSharedVariable = ulCounter;
		if( *pulSharedVariable != ulCounter )
		{
			sError = pdTRUE;
		}
	}

	/* Release the semaphore, and if no errors have occurred increment the check
	variable. */
	if(	xSemaphoreGive( pxParameters->xSemaphore ) == pdFALSE )
	{
		sError = pdTRUE;
	}

	if( sError == pdFALSE )
	{
		if( sCheckVariableToUse < semtstNUM_TASKS )
		{
			( sCheckVariables[ sCheckVariableToUse ] )++;
		}
	}

	/* If we have a block time then we are running at a priority higher
	than the idle priority.  This task takes a long time to complete
	a cycle (deliberately so to test the guarding) so will be starving
	out lower priority tasks.  Block for some time to allow give lower
	priority tasks some processor time. */
	vTaskDelay( pxParameters->xBlockTime * semtstDELAY_FACTOR );
}
else
```

```
        {
                if( pxParameters->xBlockTime == ( portTickType ) 0 )
                {
                        /* We have not got the semaphore yet, so no point using the
                        processor.  We are not blocking when attempting to obtain the
                        semaphore. */
                        taskYIELD();
                }
        }
       }
}
/*-----------------------------------------------------------*/

/* This is called to check that all the created tasks are still running. */
portBASE_TYPE xAreSemaphoreTasksStillRunning( void )
{
static portSHORT sLastCheckVariables[ semtstNUM_TASKS ] = { 0 };
portBASE_TYPE xTask, xReturn = pdTRUE;

       for( xTask = 0; xTask < semtstNUM_TASKS; xTask++ )
       {
        if( sLastCheckVariables[ xTask ] == sCheckVariables[ xTask ] )
        {
                xReturn = pdFALSE;
        }

        sLastCheckVariables[ xTask ] = sCheckVariables[ xTask ];
       }

       return xReturn;
}
```

BlockQ.c

```
/*
       FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

       This file is part of the FreeRTOS.org distribution.

       ....
       ***************************************************************************
       See http://www.FreeRTOS.org for documentation, latest information, license
       and contact details.  Please ensure to read the configuration and relevant
       port sections of the online documentation.
       ***************************************************************************
*/
```

```
/*
 * Creates six tasks that operate on three queues as follows:
 *
 * The first two tasks send and receive an incrementing number to/from a queue.
 * One task acts as a producer and the other as the consumer.  The consumer is a
 * higher priority than the producer and is set to block on queue reads.  The queue
 * only has space for one item - as soon as the producer posts a message on the
 * queue the consumer will unblock, pre-empt the producer, and remove the item.
 *
 * The second two tasks work the other way around.  Again the queue used only has
 * enough space for one item.  This time the consumer has a lower priority than the
 * producer.  The producer will try to post on the queue blocking when the queue is
 * full.  When the consumer wakes it will remove the item from the queue, causing
 * the producer to unblock, pre-empt the consumer, and immediately re-fill the
 * queue.
 *
 * The last two tasks use the same queue producer and consumer functions.  This time the queue has
 * enough space for lots of items and the tasks operate at the same priority.  The
 * producer will execute, placing items into the queue.  The consumer will start
 * executing when either the queue becomes full (causing the producer to block) or
 * a context switch occurs (tasks of the same priority will time slice).
 *
 */

/*

Changes from V4.0.2

	+ The second set of tasks were created the wrong way around.  This has been
	  corrected.
*/


#include <stdlib.h>

/* Scheduler include files. */
#include "FreeRTOS.h"
#include "task.h"
#include "queue.h"

/* Demo program include files. */
#include "BlockQ.h"

#define blckqSTACK_SIZE              configMINIMAL_STACK_SIZE
#define blckqNUM_TASK_SETS    ( 3 )
```

```
/* Structure used to pass parameters to the blocking queue tasks. */
typedef struct BLOCKING_QUEUE_PARAMETERS
{
	xQueueHandle xQueue;					/*< The queue to be used by the task. */
	portTickType xBlockTime;				/*< The block time to use on queue reads/
writes. */
	volatile portSHORT *psCheckVariable;	/*< Incremented on each successful cycle to check the
task is still running. */
} xBlockingQueueParameters;

/* Task function that creates an incrementing number and posts it on a queue. */
static portTASK_FUNCTION_PROTO( vBlockingQueueProducer, pvParameters );

/* Task function that removes the incrementing number from a queue and checks that
it is the expected number. */
static portTASK_FUNCTION_PROTO( vBlockingQueueConsumer, pvParameters );

/* Variables which are incremented each time an item is removed from a queue, and
found to be the expected value.
These are used to check that the tasks are still running. */
static volatile portSHORT sBlockingConsumerCount[ blckqNUM_TASK_SETS ] = { ( unsigned portSHORT ) 0,
( unsigned portSHORT ) 0, ( unsigned portSHORT ) 0 };

/* Variable which are incremented each time an item is posted on a queue.   These
are used to check that the tasks are still running. */
static volatile portSHORT sBlockingProducerCount[ blckqNUM_TASK_SETS ] = { ( unsigned portSHORT ) 0,
( unsigned portSHORT ) 0, ( unsigned portSHORT ) 0 };

/*-----------------------------------------------------------*/

void vStartBlockingQueueTasks( unsigned portBASE_TYPE uxPriority )
{
xBlockingQueueParameters *pxQueueParameters1, *pxQueueParameters2;
xBlockingQueueParameters *pxQueueParameters3, *pxQueueParameters4;
xBlockingQueueParameters *pxQueueParameters5, *pxQueueParameters6;
const unsigned portBASE_TYPE uxQueueSize1 = 1, uxQueueSize5 = 5;
const portTickType xBlockTime = ( portTickType ) 1000 / portTICK_RATE_MS;
const portTickType xDontBlock = ( portTickType ) 0;

	/* Create the first two tasks as described at the top of the file. */

	/* First create the structure used to pass parameters to the consumer tasks. */
	pxQueueParameters1 = ( xBlockingQueueParameters * )
pvPortMalloc( sizeof( xBlockingQueueParameters ) );

	/* Create the queue used by the first two tasks to pass the incrementing number.
	Pass a pointer to the queue in the parameter structure. */
	pxQueueParameters1->xQueue = xQueueCreate( uxQueueSize1, ( unsigned portBASE_TYPE )
sizeof( unsigned portSHORT ) );
```

```
	/* The consumer is created first so gets a block time as described above. */
	pxQueueParameters1->xBlockTime = xBlockTime;

	/* Pass in the variable that this task is going to increment so we can check it
	is still running. */
	pxQueueParameters1->psCheckVariable = &( sBlockingConsumerCount[ 0 ] );

	/* Create the structure used to pass parameters to the producer task. */
	pxQueueParameters2 = ( xBlockingQueueParameters * )
pvPortMalloc( sizeof( xBlockingQueueParameters ) );

	/* Pass the queue to this task also, using the parameter structure. */
	pxQueueParameters2->xQueue = pxQueueParameters1->xQueue;

	/* The producer is not going to block - as soon as it posts the consumer will
	wake and remove the item so the producer should always have room to post. */
	pxQueueParameters2->xBlockTime = xDontBlock;

	/* Pass in the variable that this task is going to increment so we can check
	it is still running. */
	pxQueueParameters2->psCheckVariable = &( sBlockingProducerCount[ 0 ] );


	/* Note the producer has a lower priority than the consumer when the tasks are
	spawned. */
	xTaskCreate( vBlockingQueueConsumer, ( signed portCHAR * ) "QConsB1", blckqSTACK_SIZE, ( void
* ) pxQueueParameters1, uxPriority, NULL );
	xTaskCreate( vBlockingQueueProducer, ( signed portCHAR * ) "QProdB2", blckqSTACK_SIZE, ( void
* ) pxQueueParameters2, tskIDLE_PRIORITY, NULL );



	/* Create the second two tasks as described at the top of the file.  This uses
	the same mechanism but reverses the task priorities. */

	pxQueueParameters3 = ( xBlockingQueueParameters * )
pvPortMalloc( sizeof( xBlockingQueueParameters ) );
	pxQueueParameters3->xQueue = xQueueCreate( uxQueueSize1, ( unsigned portBASE_TYPE )
sizeof( unsigned portSHORT ) );
	pxQueueParameters3->xBlockTime = xDontBlock;
	pxQueueParameters3->psCheckVariable = &( sBlockingProducerCount[ 1 ] );

	pxQueueParameters4 = ( xBlockingQueueParameters * )
pvPortMalloc( sizeof( xBlockingQueueParameters ) );
	pxQueueParameters4->xQueue = pxQueueParameters3->xQueue;
	pxQueueParameters4->xBlockTime = xBlockTime;
	pxQueueParameters4->psCheckVariable = &( sBlockingConsumerCount[ 1 ] );
```

```
	xTaskCreate( vBlockingQueueProducer, ( signed portCHAR * ) "QProdB3", blckqSTACK_SIZE, ( void * ) pxQueueParameters3, tskIDLE_PRIORITY, NULL );
	xTaskCreate( vBlockingQueueConsumer, ( signed portCHAR * ) "QConsB4", blckqSTACK_SIZE, ( void * ) pxQueueParameters4, uxPriority, NULL );



	/* Create the last two tasks as described above.  The mechanism is again just
	the same.  This time both parameter structures are given a block time. */
	pxQueueParameters5 = ( xBlockingQueueParameters * ) pvPortMalloc( sizeof( xBlockingQueueParameters ) );
	pxQueueParameters5->xQueue = xQueueCreate( uxQueueSize5, ( unsigned portBASE_TYPE ) sizeof( unsigned portSHORT ) );
	pxQueueParameters5->xBlockTime = xBlockTime;
	pxQueueParameters5->psCheckVariable = &( sBlockingProducerCount[ 2 ] );

	pxQueueParameters6 = ( xBlockingQueueParameters * ) pvPortMalloc( sizeof( xBlockingQueueParameters ) );
	pxQueueParameters6->xQueue = pxQueueParameters5->xQueue;
	pxQueueParameters6->xBlockTime = xBlockTime;
	pxQueueParameters6->psCheckVariable = &( sBlockingConsumerCount[ 2 ] );

	xTaskCreate( vBlockingQueueProducer, ( signed portCHAR * ) "QProdB5", blckqSTACK_SIZE, ( void * ) pxQueueParameters5, tskIDLE_PRIORITY, NULL );
	xTaskCreate( vBlockingQueueConsumer, ( signed portCHAR * ) "QConsB6", blckqSTACK_SIZE, ( void * ) pxQueueParameters6, tskIDLE_PRIORITY, NULL );
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vBlockingQueueProducer, pvParameters )
{
unsigned portSHORT usValue = 0;
xBlockingQueueParameters *pxQueueParameters;
portSHORT sErrorEverOccurred = pdFALSE;

	pxQueueParameters = ( xBlockingQueueParameters * ) pvParameters;

	for( ;; )
	{
		if( xQueueSend( pxQueueParameters->xQueue, ( void * ) &usValue, pxQueueParameters->xBlockTime ) != pdPASS )
		{
			sErrorEverOccurred = pdTRUE;
		}
		else
		{
			/* We have successfully posted a message, so increment the variable
			used to check we are still running. */
			if( sErrorEverOccurred == pdFALSE )
			{
```

```
                    ( *pxQueueParameters->psCheckVariable )++;
                }

                /* Increment the variable we are going to post next time round.  The
                consumer will expect the numbers to  follow in numerical order. */
                ++usValue;
        }
        }
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vBlockingQueueConsumer, pvParameters )
{
unsigned portSHORT usData, usExpectedValue = 0;
xBlockingQueueParameters *pxQueueParameters;
portSHORT sErrorEverOccurred = pdFALSE;

        pxQueueParameters = ( xBlockingQueueParameters * ) pvParameters;

        for( ;; )
        {
         if( xQueueReceive( pxQueueParameters->xQueue, &usData, pxQueueParameters->xBlockTime ) ==
pdPASS )
         {
                if( usData != usExpectedValue )
                {
                        /* Catch-up. */
                        usExpectedValue = usData;

                        sErrorEverOccurred = pdTRUE;
                }
                else
                {
                        /* We have successfully received a message, so increment the
                        variable used to check we are still running. */
                        if( sErrorEverOccurred == pdFALSE )
                        {
                                ( *pxQueueParameters->psCheckVariable )++;
                        }

                        /* Increment the value we expect to remove from the queue next time
                        round. */
                        ++usExpectedValue;
                }
         }
        }
}
```

```
/*-----------------------------------------------------------*/

/* This is called to check that all the created tasks are still running. */
portBASE_TYPE xAreBlockingQueuesStillRunning( void )
{
static portSHORT sLastBlockingConsumerCount[ blckqNUM_TASK_SETS ] = { ( unsigned portSHORT ) 0,
( unsigned portSHORT ) 0, ( unsigned portSHORT ) 0 };
static portSHORT sLastBlockingProducerCount[ blckqNUM_TASK_SETS ] = { ( unsigned portSHORT ) 0,
( unsigned portSHORT ) 0, ( unsigned portSHORT ) 0 };
portBASE_TYPE xReturn = pdPASS, xTasks;

	/* Not too worried about mutual exclusion on these variables as they are 16
	bits and we are only reading them. We also only care to see if they have
	changed or not.

	Loop through each check variable to and return pdFALSE if any are found not
	to have changed since the last call. */

	for( xTasks = 0; xTasks < blckqNUM_TASK_SETS; xTasks++ )
	{
	 if( sBlockingConsumerCount[ xTasks ] == sLastBlockingConsumerCount[ xTasks ]  )
	 {
		xReturn = pdFALSE;
	 }
	 sLastBlockingConsumerCount[ xTasks ] = sBlockingConsumerCount[ xTasks ];


	 if( sBlockingProducerCount[ xTasks ] == sLastBlockingProducerCount[ xTasks ]  )
	 {
		xReturn = pdFALSE;
	 }
	 sLastBlockingProducerCount[ xTasks ] = sBlockingProducerCount[ xTasks ];
	}

	return xReturn;
}
```

blocktim.c

```
/*
	FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

	This file is part of the FreeRTOS.org distribution.
	***

*/
```

```
/*
 * This file contains some test scenarios that ensure tasks do not exit queue
 * send or receive functions prematurely.  A description of the tests is
 * included within the code.
 */

/* Kernel includes. */
#include "FreeRTOS.h"
#include "task.h"
#include "queue.h"

/* Task priorities. */
#define bktPRIMARY_PRIORITY                 ( 3 )
#define bktSECONDARY_PRIORITY        ( 2 )

/* Task behaviour. */
#define bktQUEUE_LENGTH                             ( 5 )
#define bktSHORT_WAIT                       ( ( ( portTickType ) 20 ) / portTICK_RATE_MS )
#define bktPRIMARY_BLOCK_TIME        ( 10 )
#define bktALLOWABLE_MARGIN                 ( 12 )
#define bktTIME_TO_BLOCK                    ( 175 )
#define bktDONT_BLOCK                       ( ( portTickType ) 0 )
#define bktRUN_INDICATOR                    ( ( unsigned portBASE_TYPE ) 0x55 )

/* The queue on which the tasks block. */
static xQueueHandle xTestQueue;

/* Handle to the secondary task is required by the primary task for calls
to vTaskSuspend/Resume(). */
static xTaskHandle xSecondary;

/* Used to ensure that tasks are still executing without error. */
static portBASE_TYPE xPrimaryCycles = 0, xSecondaryCycles = 0;
static portBASE_TYPE xErrorOccurred = pdFALSE;

/* Provides a simple mechanism for the primary task to know when the
secondary task has executed. */
static volatile unsigned portBASE_TYPE xRunIndicator;

/* The two test tasks.  Their behaviour is commented within the files. */
static void vPrimaryBlockTimeTestTask( void *pvParameters );
static void vSecondaryBlockTimeTestTask( void *pvParameters );

/*-----------------------------------------------------------*/

void vCreateBlockTimeTasks( void )
```

```
{
	/* Create the queue on which the two tasks block. */
    xTestQueue = xQueueCreate( bktQUEUE_LENGTH, sizeof( portBASE_TYPE ) );

	/* Create the two test tasks. */
	xTaskCreate( vPrimaryBlockTimeTestTask, "BTest1", configMINIMAL_STACK_SIZE, NULL,
bktPRIMARY_PRIORITY, NULL );
	xTaskCreate( vSecondaryBlockTimeTestTask, "BTest2", configMINIMAL_STACK_SIZE, NULL,
bktSECONDARY_PRIORITY, &xSecondary );
}
/*-----------------------------------------------------------*/

static void vPrimaryBlockTimeTestTask( void *pvParameters )
{
portBASE_TYPE xItem, xData;
portTickType xTimeWhenBlocking;
portTickType xTimeToBlock, xBlockedTime;

	for( ;; )
	{
	 /*********************************************************************
	  Test 1

	  Simple block time wakeup test on queue receives. */
	 for( xItem = 0; xItem < bktQUEUE_LENGTH; xItem++ )
	 {
		/* The queue is empty. Attempt to read from the queue using a block
		time.  When we wake, ensure the delta in time is as expected. */
		xTimeToBlock = bktPRIMARY_BLOCK_TIME << xItem;

		/* A critical section is used to minimise the jitter in the time
		measurements. */
		portENTER_CRITICAL();
		{
			xTimeWhenBlocking = xTaskGetTickCount();

			/* We should unblock after xTimeToBlock having not received
			anything on the queue. */
			if( xQueueReceive( xTestQueue, &xData, xTimeToBlock ) != errQUEUE_EMPTY )
			{
				xErrorOccurred = pdTRUE;
			}

			/* How long were we blocked for? */
			xBlockedTime = xTaskGetTickCount() - xTimeWhenBlocking;
		}
		portEXIT_CRITICAL();
```

```
        if( xBlockedTime < xTimeToBlock )
        {
                /* Should not have blocked for less than we requested. */
                xErrorOccurred = pdTRUE;
        }

        if( xBlockedTime > ( xTimeToBlock + bktALLOWABLE_MARGIN ) )
        {
                /* Should not have blocked for longer than we requested,
                although we would not necessarily run as soon as we were
                unblocked so a margin is allowed. */
                xErrorOccurred = pdTRUE;
        }
}

/*********************************************************************
 Test 2

 Simple block time wakeup test on queue sends.

First fill the queue.  It should be empty so all sends should pass. */
for( xItem = 0; xItem < bktQUEUE_LENGTH; xItem++ )
{
        if( xQueueSend( xTestQueue, &xItem, bktDONT_BLOCK ) != pdPASS )
        {
                xErrorOccurred = pdTRUE;
        }
}

for( xItem = 0; xItem < bktQUEUE_LENGTH; xItem++ )
{
        /* The queue is full. Attempt to write to the queue using a block
        time.  When we wake, ensure the delta in time is as expected. */
        xTimeToBlock = bktPRIMARY_BLOCK_TIME << xItem;

        portENTER_CRITICAL();
        {
                xTimeWhenBlocking = xTaskGetTickCount();

                /* We should unblock after xTimeToBlock having not received
                anything on the queue. */
                if( xQueueSend( xTestQueue, &xItem, xTimeToBlock ) != errQUEUE_FULL )
                {
                        xErrorOccurred = pdTRUE;
                }
```

```
                /* How long were we blocked for? */
                xBlockedTime = xTaskGetTickCount() - xTimeWhenBlocking;
        }
        portEXIT_CRITICAL();

        if( xBlockedTime < xTimeToBlock )
        {
                /* Should not have blocked for less than we requested. */
                xErrorOccurred = pdTRUE;
        }

        if( xBlockedTime > ( xTimeToBlock + bktALLOWABLE_MARGIN ) )
        {
                /* Should not have blocked for longer than we requested,
                although we would not necessarily run as soon as we were
                unblocked so a margin is allowed. */
                xErrorOccurred = pdTRUE;
        }
}


/*********************************************************************
 Test 3

Wake the other task, it will block attempting to post to the queue.
When we read from the queue the other task will wake, but before it
can run we will post to the queue again.  When the other task runs it
will find the queue still full, even though it was woken.  It should
recognise that its block time has not expired and return to block for
the remains of its block time.

Wake the other task so it blocks attempting to post to the already
full queue. */
xRunIndicator = 0;
vTaskResume( xSecondary );

/* We need to wait a little to ensure the other task executes. */
while( xRunIndicator != bktRUN_INDICATOR )
{
        /* The other task has not yet executed. */
        vTaskDelay( bktSHORT_WAIT );
}
/* Make sure the other task is blocked on the queue. */
vTaskDelay( bktSHORT_WAIT );
xRunIndicator = 0;
```

```
for( xItem = 0; xItem < bktQUEUE_LENGTH; xItem++ )
{
	/* Now when we make space on the queue the other task should wake
	but not execute as this task has higher priority. */
	if( xQueueReceive( xTestQueue, &xData, bktDONT_BLOCK ) != pdPASS )
	{
		xErrorOccurred = pdTRUE;
	}

	/* Now fill the queue again before the other task gets a chance to
	execute.  If the other task had executed we would find the queue
	full ourselves, and the other task have set xRunIndicator. */
	if( xQueueSend( xTestQueue, &xItem, bktDONT_BLOCK ) != pdPASS )
	{
		xErrorOccurred = pdTRUE;
	}

	if( xRunIndicator == bktRUN_INDICATOR )
	{
		/* The other task should not have executed. */
		xErrorOccurred = pdTRUE;
	}

	/* Raise the priority of the other task so it executes and blocks
	on the queue again. */
	vTaskPrioritySet( xSecondary, bktPRIMARY_PRIORITY + 2 );

	/* The other task should now have re-blocked without exiting the
	queue function. */
	if( xRunIndicator == bktRUN_INDICATOR )
	{
		/* The other task should not have executed outside of the
		queue function. */
		xErrorOccurred = pdTRUE;
	}

	/* Set the priority back down. */
	vTaskPrioritySet( xSecondary, bktSECONDARY_PRIORITY );
}

/* Let the other task timeout.  When it unblockes it will check that it
unblocked at the correct time, then suspend itself. */
while( xRunIndicator != bktRUN_INDICATOR )
{
	vTaskDelay( bktSHORT_WAIT );
```

```
}
vTaskDelay( bktSHORT_WAIT );
xRunIndicator = 0;


/*****************************************************************
 Test 4

As per test 3 - but with the send and receive the other way around.
The other task blocks attempting to read from the queue.

Empty the queue.  We should find that it is full. */
for( xItem = 0; xItem < bktQUEUE_LENGTH; xItem++ )
{
	if( xQueueReceive( xTestQueue, &xData, bktDONT_BLOCK ) != pdPASS )
	{
		xErrorOccurred = pdTRUE;
	}
}

/* Wake the other task so it blocks attempting to read from  the
already empty queue. */
vTaskResume( xSecondary );

/* We need to wait a little to ensure the other task executes. */
while( xRunIndicator != bktRUN_INDICATOR )
{
	vTaskDelay( bktSHORT_WAIT );
}
vTaskDelay( bktSHORT_WAIT );
xRunIndicator = 0;

for( xItem = 0; xItem < bktQUEUE_LENGTH; xItem++ )
{
	/* Now when we place an item on the queue the other task should
	wake but not execute as this task has higher priority. */
	if( xQueueSend( xTestQueue, &xItem, bktDONT_BLOCK ) != pdPASS )
	{
		xErrorOccurred = pdTRUE;
	}

	/* Now empty the queue again before the other task gets a chance to
	execute.  If the other task had executed we would find the queue
	empty ourselves, and the other task would be suspended. */
	if( xQueueReceive( xTestQueue, &xData, bktDONT_BLOCK ) != pdPASS )
	{
```

```
				xErrorOccurred = pdTRUE;
			}

			if( xRunIndicator == bktRUN_INDICATOR )
			{
				/* The other task should not have executed. */
				xErrorOccurred = pdTRUE;
			}

			/* Raise the priority of the other task so it executes and blocks
			on the queue again. */
			vTaskPrioritySet( xSecondary, bktPRIMARY_PRIORITY + 2 );

			/* The other task should now have re-blocked without exiting the
			queue function. */
			if( xRunIndicator == bktRUN_INDICATOR )
			{
				/* The other task should not have executed outside of the
				queue function. */
				xErrorOccurred = pdTRUE;
			}
			vTaskPrioritySet( xSecondary, bktSECONDARY_PRIORITY );
		}

		/* Let the other task timeout.  When it unblockes it will check that it
		unblocked at the correct time, then suspend itself. */
		while( xRunIndicator != bktRUN_INDICATOR )
		{
			vTaskDelay( bktSHORT_WAIT );
		}
		vTaskDelay( bktSHORT_WAIT );

		xPrimaryCycles++;
	}
}
/*-----------------------------------------------------------*/

static void vSecondaryBlockTimeTestTask( void *pvParameters )
{
portTickType xTimeWhenBlocking, xBlockedTime;
portBASE_TYPE xData;

	for( ;; )
	{
		/*********************************************************************
		 Test 1 and 2
```

```
This task does does not participate in these tests. */
vTaskSuspend( NULL );

/*****************************************************************
 Test 3

The first thing we do is attempt to read from the queue.  It should be
full so we block.  Note the time before we block so we can check the
wake time is as per that expected. */
portENTER_CRITICAL();
{
	xTimeWhenBlocking = xTaskGetTickCount();

	/* We should unblock after bktTIME_TO_BLOCK having not received
	anything on the queue. */
	xData = 0;
	xRunIndicator = bktRUN_INDICATOR;
	if( xQueueSend( xTestQueue, &xData, bktTIME_TO_BLOCK ) != errQUEUE_FULL )
	{
		xErrorOccurred = pdTRUE;
	}

	/* How long were we inside the send function? */
	xBlockedTime = xTaskGetTickCount() - xTimeWhenBlocking;
}
portEXIT_CRITICAL();

/* We should not have blocked for less time than bktTIME_TO_BLOCK. */
if( xBlockedTime < bktTIME_TO_BLOCK )
{
	xErrorOccurred = pdTRUE;
}

/* We should of not blocked for much longer than bktALLOWABLE_MARGIN
either.  A margin is permitted as we would not necessarily run as
soon as we unblocked. */
if( xBlockedTime > ( bktTIME_TO_BLOCK + bktALLOWABLE_MARGIN ) )
{
	xErrorOccurred = pdTRUE;
}

/* Suspend ready for test 3. */
xRunIndicator = bktRUN_INDICATOR;
vTaskSuspend( NULL );
```

```
        /********************************************************************
         Test 4

        As per test three, but with the send and receive reversed. */
        portENTER_CRITICAL();
        {
                xTimeWhenBlocking = xTaskGetTickCount();

                /* We should unblock after bktTIME_TO_BLOCK having not received
                anything on the queue. */
                xRunIndicator = bktRUN_INDICATOR;
                if( xQueueReceive( xTestQueue, &xData, bktTIME_TO_BLOCK ) != errQUEUE_EMPTY )
                {
                        xErrorOccurred = pdTRUE;
                }

                xBlockedTime = xTaskGetTickCount() - xTimeWhenBlocking;
        }
        portEXIT_CRITICAL();

        /* We should not have blocked for less time than bktTIME_TO_BLOCK. */
        if( xBlockedTime < bktTIME_TO_BLOCK )
        {
                xErrorOccurred = pdTRUE;
        }

        /* We should of not blocked for much longer than bktALLOWABLE_MARGIN
        either.  A margin is permitted as we would not necessarily run as soon
        as we unblocked. */
        if( xBlockedTime > ( bktTIME_TO_BLOCK + bktALLOWABLE_MARGIN ) )
        {
                xErrorOccurred = pdTRUE;
        }

        xRunIndicator = bktRUN_INDICATOR;

        xSecondaryCycles++;
       }
}
/*-----------------------------------------------------------*/

portBASE_TYPE xAreBlockTimeTestTasksStillRunning( void )
{
static portBASE_TYPE xLastPrimaryCycleCount = 0, xLastSecondaryCycleCount = 0;
portBASE_TYPE xReturn = pdPASS;
```

```
	/* Have both tasks performed at least one cycle since this function was
	last called? */
	if( xPrimaryCycles == xLastPrimaryCycleCount )
	{
		xReturn = pdFAIL;
	}

	if( xSecondaryCycles == xLastSecondaryCycleCount )
	{
		xReturn = pdFAIL;
	}

	if( xErrorOccurred == pdTRUE )
	{
		xReturn = pdFAIL;
	}

	xLastSecondaryCycleCount = xSecondaryCycles;
	xLastPrimaryCycleCount = xPrimaryCycles;

	return xReturn;
}
```

comtest.c

```
/*
	FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

	This file is part of the FreeRTOS.org distribution.

	***
*/



/*
 * This version of comtest. c is for use on systems that have limited stack
 * space and no display facilities.  The complete version can be found in
 * the Demo/Common/Full directory.
 *
 * Creates two tasks that operate on an interrupt driven serial port.  A
 * loopback connector should be used so that everything that is transmitted is
 * also received.  The serial port does not use any flow control.  On a
 * standard 9way 'D' connector pins two and three should be connected together.
 *
 * The first task posts a sequence of characters to the Tx queue, toggling an
 * LED on each successful post.  At the end of the sequence it sleeps for a
```

```
 * pseudo-random period before resending the same sequence.
 *
 * The UART Tx end interrupt is enabled whenever data is available in the Tx
 * queue.  The Tx end ISR removes a single character from the Tx queue and
 * passes it to the UART for transmission.
 *
 * The second task blocks on the Rx queue waiting for a character to become
 * available.  When the UART Rx end interrupt receives a character it places
 * it in the Rx queue, waking the second task.  The second task checks that the
 * characters removed from the Rx queue form the same sequence as those posted
 * to the Tx queue, and toggles an LED for each correct character.
 *
 * The receiving task is spawned with a higher priority than the transmitting
 * task.  The receiver will therefore wake every time a character is
 * transmitted so neither the Tx or Rx queue should ever hold more than a few
 * characters.
 *
 */

/*
Changes from V1.2.0:

	+ Reduced the maximum time between successive transmissions.  This provides
	  for a more rigorous test.

Changes from V2.0.0

	+ Delay periods are now specified using variables and constants of
	  portTickType rather than unsigned portLONG.

Changes from V2.5.1

	+ The constant comOFFSET_TIME added to the delay period to ensure a more
	  random delay period is used.
*/

/* Scheduler include files. */
#include <stdlib.h>
#include "FreeRTOS.h"
#include "task.h"

/* Demo program include files. */
#include "queue.h" // required by serial.h
#include "serial.h"
#include "comtest.h"
#include "partest.h"
```

```
#define comSTACK_SIZE                       configMINIMAL_STACK_SIZE
#define comTX_LED_OFFSET                    ( 0 )
#define comRX_LED_OFFSET                    ( 1 )
#define comTOTAL_PERMISSIBLE_ERRORS ( 2 )

/* The Tx task will transmit the sequence of characters at a pseudo random
interval.  This is the maximum and minimum block time between sends. */
#define comTX_MAX_BLOCK_TIME         ( ( portTickType ) 0x96 )
#define comTX_MIN_BLOCK_TIME         ( ( portTickType ) 0x32 )
#define comOFFSET_TIME                      ( ( portTickType ) 3 )

/* We should find that each character can be queued for Tx immediately and we
don't have to block to send. */
#define comNO_BLOCK                                 ( ( portTickType ) 0 )

/* The Rx task will block on the Rx queue for a long period. */
#define comRX_BLOCK_TIME                    ( ( portTickType ) 0xffff )

/* The sequence transmitted is from comFIRST_BYTE to and including comLAST_BYTE. */
#define comFIRST_BYTE                       ( 'A' )
#define comLAST_BYTE                        ( 'X' )

#define comBUFFER_LEN                       ( ( unsigned portBASE_TYPE ) ( comLAST_BYTE -
comFIRST_BYTE ) + ( unsigned portBASE_TYPE ) 1 )
#define comINITIAL_RX_COUNT_VALUE    ( 0 )

/* Handle to the com port used by both tasks. */
static xComPortHandle xPort = NULL;

/* The transmit task as described at the top of the file. */
static portTASK_FUNCTION_PROTO( vComTxTask, pvParameters );

/* The receive task as described at the top of the file. */
static portTASK_FUNCTION_PROTO( vComRxTask, pvParameters );

/* The LED that should be toggled by the Rx and Tx tasks.  The Rx task will
toggle LED ( uxBaseLED + comRX_LED_OFFSET).  The Tx task will toggle LED
( uxBaseLED + comTX_LED_OFFSET ). */
static unsigned portBASE_TYPE uxBaseLED = 0;

/* Check variable used to ensure no error have occurred.  The Rx task will
increment this variable after every successfully received sequence.  If at any
time the sequence is incorrect the the variable will stop being incremented. */
static volatile unsigned portBASE_TYPE uxRxLoops = comINITIAL_RX_COUNT_VALUE;

/*-----------------------------------------------------------*/
```

```
void vAltStartComTestTasks( unsigned portBASE_TYPE uxPriority, unsigned portLONG ulBaudRate,
unsigned portBASE_TYPE uxLED )
{
	/* Initialise the com port then spawn the Rx and Tx tasks. */
	uxBaseLED = uxLED;
	xSerialPortInitMinimal( ulBaudRate, comBUFFER_LEN );

	/* The Tx task is spawned with a lower priority than the Rx task. */
	xTaskCreate( vComTxTask, ( const signed portCHAR * const ) "COMTx", comSTACK_SIZE, NULL,
uxPriority - 1, ( xTaskHandle * ) NULL );
	xTaskCreate( vComRxTask, ( const signed portCHAR * const ) "COMRx", comSTACK_SIZE, NULL,
uxPriority, ( xTaskHandle * ) NULL );
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vComTxTask, pvParameters )
{
signed portCHAR cByteToSend;
portTickType xTimeToWait;

	/* Just to stop compiler warnings. */
	( void ) pvParameters;

	for( ;; )
	{
		/* Simply transmit a sequence of characters from comFIRST_BYTE to
		comLAST_BYTE. */
		for( cByteToSend = comFIRST_BYTE; cByteToSend <= comLAST_BYTE; cByteToSend++ )
		{
			if( xSerialPutChar( xPort, cByteToSend, comNO_BLOCK ) == pdPASS )
			{
				vParTestToggleLED( uxBaseLED + comTX_LED_OFFSET );
			}
		}

		/* Turn the LED off while we are not doing anything. */
		vParTestSetLED( uxBaseLED + comTX_LED_OFFSET, pdFALSE );

		/* We have posted all the characters in the string - wait before
		re-sending.  Wait a pseudo-random time as this will provide a better
		test. */
		xTimeToWait = xTaskGetTickCount() + comOFFSET_TIME;

		/* Make sure we don't wait too long... */
		xTimeToWait %= comTX_MAX_BLOCK_TIME;

		/* ...but we do want to wait. */
```

```
		if( xTimeToWait < comTX_MIN_BLOCK_TIME )
		{
			xTimeToWait = comTX_MIN_BLOCK_TIME;
		}

		vTaskDelay( xTimeToWait );
	}
} /*lint !e715 !e818 pvParameters is required for a task function even if it is not referenced. */
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vComRxTask, pvParameters )
{
signed portCHAR cExpectedByte, cByteRxed;
portBASE_TYPE xResyncRequired = pdFALSE, xErrorOccurred = pdFALSE;

	/* Just to stop compiler warnings. */
	( void ) pvParameters;

	for( ;; )
	{
		/* We expect to receive the characters from comFIRST_BYTE to
		comLAST_BYTE in an incrementing order.  Loop to receive each byte. */
		for( cExpectedByte = comFIRST_BYTE; cExpectedByte <= comLAST_BYTE; cExpectedByte++ )
		{
			/* Block on the queue that contains received bytes until a byte is
			available. */
			if( xSerialGetChar( xPort, &cByteRxed, comRX_BLOCK_TIME ) )
			{
				/* Was this the byte we were expecting?  If so, toggle the LED,
				otherwise we are out on sync and should break out of the loop
				until the expected character sequence is about to restart. */
				if( cByteRxed == cExpectedByte )
				{
					vParTestToggleLED( uxBaseLED + comRX_LED_OFFSET );
				}
				else
				{
					xResyncRequired = pdTRUE;
					break; /*lint !e960 Non-switch break allowed. */
				}
			}
		}

		/* Turn the LED off while we are not doing anything. */
		vParTestSetLED( uxBaseLED + comRX_LED_OFFSET, pdFALSE );
```

```
		/* Did we break out of the loop because the characters were received in
		an unexpected order?  If so wait here until the character sequence is
		about to restart. */
		if( xResyncRequired == pdTRUE )
		{
			while( cByteRxed != comLAST_BYTE )
			{
				/* Block until the next char is available. */
				xSerialGetChar( xPort, &cByteRxed, comRX_BLOCK_TIME );
			}

			/* Note that an error occurred which caused us to have to resync.
			We use this to stop incrementing the loop counter so
			sAreComTestTasksStillRunning() will return false - indicating an
			error. */
			xErrorOccurred++;

			/* We have now resynced with the Tx task and can continue. */
			xResyncRequired = pdFALSE;
		}
		else
		{
			if( xErrorOccurred < comTOTAL_PERMISSIBLE_ERRORS )
			{
				/* Increment the count of successful loops.  As error
				occurring (i.e. an unexpected character being received) will
				prevent this counter being incremented for the rest of the
				execution.   Don't worry about mutual exclusion on this
				variable - it doesn't really matter as we just want it
				to change. */
				uxRxLoops++;
			}
		}
	}
} /*lint !e715 !e818 pvParameters is required for a task function even if it is not referenced. */
/*-----------------------------------------------------------*/

portBASE_TYPE xAreComTestTasksStillRunning( void )
{
portBASE_TYPE xReturn;

	/* If the count of successful reception loops has not changed then at
	some time an error occurred (i.e. a character was received out of sequence)
	and we will return false. */
	if( uxRxLoops == comINITIAL_RX_COUNT_VALUE )
	{
```

```
        xReturn = pdFALSE;
        }
        else
        {
        xReturn = pdTRUE;
        }

        /* Reset the count of successful Rx loops.  When this function is called
        again we expect this to have been incremented. */
        uxRxLoops = comINITIAL_RX_COUNT_VALUE;

        return xReturn;
}
```

death.c

```
/*
        FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

        This file is part of the FreeRTOS.org distribution.

        ***
*/

/**
 * Create a single persistent task which periodically dynamically creates another
 * four tasks.  The original task is called the creator task, the four tasks it
 * creates are called suicidal tasks.
 *
 * Two of the created suicidal tasks kill one other suicidal task before killing
 * themselves - leaving just the original task remaining.
 *
 * The creator task must be spawned after all of the other demo application tasks
 * as it keeps a check on the number of tasks under the scheduler control.  The
 * number of tasks it expects to see running should never be greater than the
 * number of tasks that were in existence when the creator task was spawned, plus
 * one set of four suicidal tasks.  If this number is exceeded an error is flagged.
 *
 * \page DeathC death.c
 * \ingroup DemoFiles
 * <HR>
 */

/*
Changes from V3.0.0
        + CreationCount sizes changed from unsigned portBASE_TYPE to
          unsigned portSHORT to minimize the risk of overflowing.
```

```
	+ Reset of usLastCreationCount added

Changes from V3.1.0
	+ Changed the dummy calculation to use variables of type long, rather than
	  float.  This allows the file to be used with ports that do not support
	  floating point.

*/

#include <stdlib.h>

/* Scheduler include files. */
#include "FreeRTOS.h"
#include "task.h"

/* Demo program include files. */
#include "death.h"

#define deathSTACK_SIZE		( configMINIMAL_STACK_SIZE + 24 )

/* The task originally created which is responsible for periodically dynamically
creating another four tasks. */
static portTASK_FUNCTION_PROTO( vCreateTasks, pvParameters );

/* The task function of the dynamically created tasks. */
static portTASK_FUNCTION_PROTO( vSuicidalTask, pvParameters );

/* A variable which is incremented every time the dynamic tasks are created.  This
is used to check that the task is still running. */
static volatile unsigned portSHORT usCreationCount = 0;

/* Used to store the number of tasks that were originally running so the creator
task can tell if any of the suicidal tasks have failed to die.
*/
static volatile unsigned portBASE_TYPE uxTasksRunningAtStart = 0;

/* Tasks are deleted by the idle task.  Under heavy load the idle task might
not get much processing time, so it would be legitimate for several tasks to
remain undeleted for a short period. */
static const unsigned portBASE_TYPE uxMaxNumberOfExtraTasksRunning = 4;

/* Used to store a handle to the tasks that should be killed by a suicidal task,
before it kills itself. */
xTaskHandle xCreatedTask1, xCreatedTask2;
```

```
/*-----------------------------------------------------------*/

void vCreateSuicidalTasks( unsigned portBASE_TYPE uxPriority )
{
unsigned portBASE_TYPE *puxPriority;

	/* Create the Creator tasks - passing in as a parameter the priority at which
	the suicidal tasks should be created. */
	puxPriority = ( unsigned portBASE_TYPE * ) pvPortMalloc( sizeof( unsigned portBASE_TYPE ) );
	*puxPriority = uxPriority;

	xTaskCreate( vCreateTasks, "CREATOR", deathSTACK_SIZE, ( void * ) puxPriority, uxPriority,
NULL );

	/* Record the number of tasks that are running now so we know if any of the
	suicidal tasks have failed to be killed. */
	uxTasksRunningAtStart = ( unsigned portBASE_TYPE ) uxTaskGetNumberOfTasks();

	/* FreeRTOS.org versions before V3.0 started the idle-task as the very
	first task. The idle task was then already included in uxTasksRunningAtStart.
	From FreeRTOS V3.0 on, the idle task is started when the scheduler is
	started. Therefore the idle task is not yet accounted for. We correct
	this by increasing uxTasksRunningAtStart by 1. */
	uxTasksRunningAtStart++;
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vSuicidalTask, pvParameters )
{
volatile portLONG l1, l2;
xTaskHandle xTaskToKill;
const portTickType xDelay = ( portTickType ) 200 / portTICK_RATE_MS;

	if( pvParameters != NULL )
	{
	 /* This task is periodically created four times.  Two created tasks are
	 passed a handle to the other task so it can kill it before killing itself.
	 The other task is passed in null. */
	 xTaskToKill = *( xTaskHandle* )pvParameters;
	}
	else
	{
	 xTaskToKill = NULL;
	}

	for( ;; )
	{
```

```
        /* Do something random just to use some stack and registers. */
        l1 = 2;
        l2 = 89;
        l2 *= l1;
        vTaskDelay( xDelay );

        if( xTaskToKill != NULL )
        {
                /* Make sure the other task has a go before we delete it. */
                vTaskDelay( ( portTickType ) 0 );
                /* Kill the other task that was created by vCreateTasks(). */
                vTaskDelete( xTaskToKill );
                /* Kill ourselves. */
                vTaskDelete( NULL );
        }
       }
}/*lint !e818 !e550 Function prototype must be as per standard for task functions. */
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vCreateTasks, pvParameters )
{
const portTickType xDelay = ( portTickType ) 1000 / portTICK_RATE_MS;
unsigned portBASE_TYPE uxPriority;

       uxPriority = *( unsigned portBASE_TYPE * ) pvParameters;
       vPortFree( pvParameters );

       for( ;; )
       {
        /* Just loop round, delaying then creating the four suicidal tasks. */
        vTaskDelay( xDelay );

        xTaskCreate( vSuicidalTask, "SUICID1", deathSTACK_SIZE, NULL, uxPriority, &xCreatedTask1 );
        xTaskCreate( vSuicidalTask, "SUICID2", deathSTACK_SIZE, &xCreatedTask1, uxPriority, NULL );

        xTaskCreate( vSuicidalTask, "SUICID1", deathSTACK_SIZE, NULL, uxPriority, &xCreatedTask2 );
        xTaskCreate( vSuicidalTask, "SUICID2", deathSTACK_SIZE, &xCreatedTask2, uxPriority, NULL );

        ++usCreationCount;
       }
}
/*-----------------------------------------------------------*/

/* This is called to check that the creator task is still running and that there
are not any more than four extra tasks. */
portBASE_TYPE xIsCreateTaskStillRunning( void )
```

```
{
static portSHORT usLastCreationCount = -1;
portBASE_TYPE xReturn = pdTRUE;
static unsigned portBASE_TYPE uxTasksRunningNow;

        if( usLastCreationCount == usCreationCount )
        {
         xReturn = pdFALSE;
        }
        else
        {
         usLastCreationCount = usCreationCount;
        }

        uxTasksRunningNow = ( unsigned portBASE_TYPE ) uxTaskGetNumberOfTasks();

        if( uxTasksRunningNow < uxTasksRunningAtStart )
        {
         xReturn = pdFALSE;
        }
        else if( ( uxTasksRunningNow - uxTasksRunningAtStart ) > uxMaxNumberOfExtraTasksRunning )
        {
         xReturn = pdFALSE;
        }
        else
        {
         /* Everything is okay. */
        }

        return xReturn;
}
```

dynamic.c

```
/*
        FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

        This file is part of the FreeRTOS.org distribution.
        ***
 */

/*
 * The first test creates three tasks - two counter tasks (one continuous count
 * and one limited count) and one controller.  A "count" variable is shared
 * between all three tasks.  The two counter tasks should never be in a "ready"
 * state at the same time.  The controller task runs at the same priority as
```

```
* the continuous count task, and at a lower priority than the limited count
* task.
*
* One counter task loops indefinitely, incrementing the shared count variable
* on each iteration.  To ensure it has exclusive access to the variable it
* raises it's priority above that of the controller task before each
* increment, lowering it again to it's original priority before starting the
* next iteration.
*
* The other counter task increments the shared count variable on each
* iteration of it's loop until the count has reached a limit of 0xff - at
* which point it suspends itself.  It will not start a new loop until the
* controller task has made it "ready" again by calling vTaskResume ().
* This second counter task operates at a higher priority than controller
* task so does not need to worry about mutual exclusion of the counter
* variable.
*
* The controller task is in two sections.  The first section controls and
* monitors the continuous count task.  When this section is operational the
* limited count task is suspended.  Likewise, the second section controls
* and monitors the limited count task.  When this section is operational the
* continuous count task is suspended.
*
* In the first section the controller task first takes a copy of the shared
* count variable.  To ensure mutual exclusion on the count variable it
* suspends the continuous count task, resuming it again when the copy has been
* taken.  The controller task then sleeps for a fixed period - during which
* the continuous count task will execute and increment the shared variable.
* When the controller task wakes it checks that the continuous count task
* has executed by comparing the copy of the shared variable with its current
* value.  This time, to ensure mutual exclusion, the scheduler itself is
* suspended with a call to vTaskSuspendAll ().  This is for demonstration
* purposes only and is not a recommended technique due to its inefficiency.
*
* After a fixed number of iterations the controller task suspends the
* continuous count task, and moves on to its second section.
*
* At the start of the second section the shared variable is cleared to zero.
* The limited count task is then woken from it's suspension by a call to
* vTaskResume ().  As this counter task operates at a higher priority than
* the controller task the controller task should not run again until the
* shared variable has been counted up to the limited value causing the counter
* task to suspend itself.  The next line after vTaskResume () is therefore
* a check on the shared variable to ensure everything is as expected.
*
*
```

```
 * The second test consists of a couple of very simple tasks that post onto a
 * queue while the scheduler is suspended.  This test was added to test parts
 * of the scheduler not exercised by the first test.
 *
 */

#include <stdlib.h>

/* Scheduler include files. */
#include "FreeRTOS.h"
#include "task.h"
#include "semphr.h"

/* Demo app include files. */
#include "dynamic.h"

/* Function that implements the "limited count" task as described above. */
static portTASK_FUNCTION_PROTO( vLimitedIncrementTask, pvParameters );

/* Function that implements the "continuous count" task as described above. */
static portTASK_FUNCTION_PROTO( vContinuousIncrementTask, pvParameters );

/* Function that implements the controller task as described above. */
static portTASK_FUNCTION_PROTO( vCounterControlTask, pvParameters );

static portTASK_FUNCTION_PROTO( vQueueReceiveWhenSuspendedTask, pvParameters );
static portTASK_FUNCTION_PROTO( vQueueSendWhenSuspendedTask, pvParameters );

/* Demo task specific constants. */
#define priSTACK_SIZE                       ( ( unsigned portSHORT ) 128 )
#define priSLEEP_TIME                       ( ( portTickType ) 100 )
#define priLOOPS                                    ( 5 )
#define priMAX_COUNT                        ( ( unsigned portLONG ) 0xff )
#define priNO_BLOCK                                 ( ( portTickType ) 0 )
#define priSUSPENDED_QUEUE_LENGTH    ( 1 )

/*-----------------------------------------------------------*/

/* Handles to the two counter tasks.  These could be passed in as parameters
to the controller task to prevent them having to be file scope. */
static xTaskHandle xContinousIncrementHandle, xLimitedIncrementHandle;

/* The shared counter variable.  This is passed in as a parameter to the two
counter variables for demonstration purposes. */
static unsigned portLONG ulCounter;
```

```
/* Variables used to check that the tasks are still operating without error.
Each complete iteration of the controller task increments this variable
provided no errors have been found.  The variable maintaining the same value
is therefore indication of an error. */
static unsigned portSHORT usCheckVariable = ( unsigned portSHORT ) 0;
static portBASE_TYPE xSuspendedQueueSendError = pdFALSE;
static portBASE_TYPE xSuspendedQueueReceiveError = pdFALSE;

/* Queue used by the second test. */
xQueueHandle xSuspendedTestQueue;

/*-----------------------------------------------------------*/
/*
 * Start the three tasks as described at the top of the file.
 * Note that the limited count task is given a higher priority.
 */
void vStartDynamicPriorityTasks( void )
{
	xSuspendedTestQueue = xQueueCreate( priSUSPENDED_QUEUE_LENGTH, sizeof( unsigned portLONG ) );
	xTaskCreate( vContinuousIncrementTask, ( signed portCHAR * ) "CNT_INC", priSTACK_SIZE, ( void
* ) &ulCounter, tskIDLE_PRIORITY, &xContinousIncrementHandle );
	xTaskCreate( vLimitedIncrementTask, ( signed portCHAR * ) "LIM_INC", priSTACK_SIZE, ( void
* ) &ulCounter, tskIDLE_PRIORITY + 1, &xLimitedIncrementHandle );
	xTaskCreate( vCounterControlTask, ( signed portCHAR * ) "C_CTRL", priSTACK_SIZE, NULL,
tskIDLE_PRIORITY, NULL );
	xTaskCreate( vQueueSendWhenSuspendedTask, ( signed portCHAR * ) "SUSP_TX", priSTACK_SIZE,
NULL, tskIDLE_PRIORITY, NULL );
	xTaskCreate( vQueueReceiveWhenSuspendedTask, ( signed portCHAR * ) "SUSP_RX", priSTACK_SIZE,
NULL, tskIDLE_PRIORITY, NULL );
}
/*-----------------------------------------------------------*/

/*
 * Just loops around incrementing the shared variable until the limit has been
 * reached.  Once the limit has been reached it suspends itself.
 */
static portTASK_FUNCTION( vLimitedIncrementTask, pvParameters )
{
unsigned portLONG *pulCounter;

	/* Take a pointer to the shared variable from the parameters passed into
	the task. */
	pulCounter = ( unsigned portLONG * ) pvParameters;

	/* This will run before the control task, so the first thing it does is
	suspend - the control task will resume it when ready. */
	vTaskSuspend( NULL );
```

```
	for( ;; )
	{
	 /* Just count up to a value then suspend. */
	 ( *pulCounter )++;

	 if( *pulCounter >= priMAX_COUNT )
	 {
		vTaskSuspend( NULL );
	 }
	}
}
/*-----------------------------------------------------------*/

/*
 * Just keep counting the shared variable up.  The control task will suspend
 * this task when it wants.
 */
static portTASK_FUNCTION( vContinuousIncrementTask, pvParameters )
{
unsigned portLONG *pulCounter;
unsigned portBASE_TYPE uxOurPriority;

	/* Take a pointer to the shared variable from the parameters passed into
	the task. */
	pulCounter = ( unsigned portLONG * ) pvParameters;

	/* Query our priority so we can raise it when exclusive access to the
	shared variable is required. */
	uxOurPriority = uxTaskPriorityGet( NULL );

	for( ;; )
	{
	 /* Raise our priority above the controller task to ensure a context
	 switch does not occur while we are accessing this variable. */
	 vTaskPrioritySet( NULL, uxOurPriority + 1 );
		( *pulCounter )++;
	 vTaskPrioritySet( NULL, uxOurPriority );
	}
}
/*-----------------------------------------------------------*/

/*
 * Controller task as described above.
 */
static portTASK_FUNCTION( vCounterControlTask, pvParameters )
{
```

```
unsigned portLONG ulLastCounter;
portSHORT sLoops;
portSHORT sError = pdFALSE;

        /* Just to stop warning messages. */
        ( void ) pvParameters;

        for( ;; )
        {
         /* Start with the counter at zero. */
         ulCounter = ( unsigned portLONG ) 0;

         /* First section : */

         /* Check the continuous count task is running. */
         for( sLoops = 0; sLoops < priLOOPS; sLoops++ )
         {
                /* Suspend the continuous count task so we can take a mirror of the
                shared variable without risk of corruption. */
                vTaskSuspend( xContinousIncrementHandle );
                        ulLastCounter = ulCounter;
                vTaskResume( xContinousIncrementHandle );

                /* Now delay to ensure the other task has processor time. */
                vTaskDelay( priSLEEP_TIME );

                /* Check the shared variable again.  This time to ensure mutual
                exclusion the whole scheduler will be locked.  This is just for
                demo purposes! */
                vTaskSuspendAll();
                {
                        if( ulLastCounter == ulCounter )
                        {
                                /* The shared variable has not changed.  There is a problem
                                with the continuous count task so flag an error. */
                                sError = pdTRUE;
                        }
                }
                xTaskResumeAll();
         }


         /* Second section: */

         /* Suspend the continuous counter task so it stops accessing the shared variable. */
         vTaskSuspend( xContinousIncrementHandle );
```

```
        /* Reset the variable. */
        ulCounter = ( unsigned portLONG ) 0;

        /* Resume the limited count task which has a higher priority than us.
        We should therefore not return from this call until the limited count
        task has suspended itself with a known value in the counter variable. */
        vTaskResume( xLimitedIncrementHandle );

        /* Does the counter variable have the expected value? */
        if( ulCounter != priMAX_COUNT )
        {
                sError = pdTRUE;
        }

        if( sError == pdFALSE )
        {
                /* If no errors have occurred then increment the check variable. */
                portENTER_CRITICAL();
                        usCheckVariable++;
                portEXIT_CRITICAL();
        }

        /* Resume the continuous count task and do it all again. */
        vTaskResume( xContinousIncrementHandle );
       }
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vQueueSendWhenSuspendedTask, pvParameters )
{
static unsigned portLONG ulValueToSend = ( unsigned portLONG ) 0;

       /* Just to stop warning messages. */
       ( void ) pvParameters;

       for( ;; )
       {
        vTaskSuspendAll();
        {
                /* We must not block while the scheduler is suspended! */
                if( xQueueSend( xSuspendedTestQueue, ( void * ) &ulValueToSend, priNO_BLOCK ) !=
pdTRUE )
                {
                        xSuspendedQueueSendError = pdTRUE;
                }
        }
```

```
        xTaskResumeAll();

        vTaskDelay( priSLEEP_TIME );

        ++ulValueToSend;
       }
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vQueueReceiveWhenSuspendedTask, pvParameters )
{
static unsigned portLONG ulExpectedValue = ( unsigned portLONG ) 0, ulReceivedValue;
portBASE_TYPE xGotValue;

       /* Just to stop warning messages. */
       ( void ) pvParameters;

       for( ;; )
       {
        do
        {
               /* Suspending the scheduler here is fairly pointless and
               undesirable for a normal application.  It is done here purely
               to test the scheduler.  The inner xTaskResumeAll() should
               never return pdTRUE as the scheduler is still locked by the
               outer call. */
               vTaskSuspendAll();
               {
                      vTaskSuspendAll();
                      {
                             xGotValue = xQueueReceive( xSuspendedTestQueue, ( void * )
&ulReceivedValue, priNO_BLOCK );
                      }
                      if( xTaskResumeAll() )
                      {
                             xSuspendedQueueReceiveError = pdTRUE;
                      }
               }
               xTaskResumeAll();

        } while( xGotValue == pdFALSE );

        if( ulReceivedValue != ulExpectedValue )
        {
               xSuspendedQueueReceiveError = pdTRUE;
        }
```

```
		++ulExpectedValue;
	}
}
/*-----------------------------------------------------------*/

/* Called to check that all the created tasks are still running without error. */
portBASE_TYPE xAreDynamicPriorityTasksStillRunning( void )
{
/* Keep a history of the check variables so we know if it has been incremented
since the last call. */
static unsigned portSHORT usLastTaskCheck = ( unsigned portSHORT ) 0;
portBASE_TYPE xReturn = pdTRUE;

	/* Check the tasks are still running by ensuring the check variable
	is still incrementing. */

	if( usCheckVariable == usLastTaskCheck )
	{
		/* The check has not incremented so an error exists. */
		xReturn = pdFALSE;
	}

	if( xSuspendedQueueSendError == pdTRUE )
	{
		xReturn = pdFALSE;
	}

	if( xSuspendedQueueReceiveError == pdTRUE )
	{
		xReturn = pdFALSE;
	}

	usLastTaskCheck = usCheckVariable;
	return xReturn;
}
```

Integer.c

```
/*
	FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

	This file is part of the FreeRTOS.org distribution.

	***
*/

/*
```

```
 * This version of integer. c is for use on systems that have limited stack
 * space and no display facilities.  The complete version can be found in
 * the Demo/Common/Full directory.
 *
 * As with the full version, the tasks created in this file are a good test
 * of the scheduler context switch mechanism.  The processor has to access
 * 32bit variables in two or four chunks (depending on the processor).  The low
 * priority of these tasks means there is a high probability that a context
 * switch will occur mid calculation.  See flop. c documentation for
 * more information.
 *
 */

/*
Changes from V1.2.1

	+ The constants used in the calculations are larger to ensure the
	  optimiser does not truncate them to 16 bits.

Changes from V1.2.3

	+ uxTaskCheck is now just used as a boolean.  Instead of incrementing
	  the variable each cycle of the task, the variable is simply set to
	  true.  sAreIntegerMathsTaskStillRunning() sets it back to false and
	  expects it to have been set back to true by the time it is called
	  again.
	+ A division has been included in the calculation.
*/

#include <stdlib.h>

/* Scheduler include files. */
#include "FreeRTOS.h"
#include "task.h"

/* Demo program include files. */
#include "integer.h"

/* The constants used in the calculation. */
#define intgCONST1				( ( portLONG ) 123 )
#define intgCONST2				( ( portLONG ) 234567 )
#define intgCONST3				( ( portLONG ) -3 )
#define intgCONST4				( ( portLONG ) 7 )
#define intgEXPECTED_ANSWER		( ( ( intgCONST1 + intgCONST2 ) * intgCONST3 ) / intgCONST4 )

#define intgSTACK_SIZE			configMINIMAL_STACK_SIZE
```

```
/* As this is the minimal version, we will only create one task. */
#define intgNUMBER_OF_TASKS          ( 1 )

/* The task function.  Repeatedly performs a 32 bit calculation, checking the
result against the expected result.  If the result is incorrect then the
context switch must have caused some corruption. */
static portTASK_FUNCTION_PROTO( vCompeteingIntMathTask, pvParameters );

/* Variables that are set to true within the calculation task to indicate
that the task is still executing.  The check task sets the variable back to
false, flagging an error if the variable is still false the next time it
is called. */
static volatile signed portBASE_TYPE xTaskCheck[ intgNUMBER_OF_TASKS ] = { ( signed portBASE_TYPE )
pdFALSE };

/*-----------------------------------------------------------*/

void vStartIntegerMathTasks( unsigned portBASE_TYPE uxPriority )
{
portSHORT sTask;

	for( sTask = 0; sTask < intgNUMBER_OF_TASKS; sTask++ )
	{
	 xTaskCreate( vCompeteingIntMathTask, ( const signed portCHAR * const ) "IntMath",
intgSTACK_SIZE, ( void * ) &( xTaskCheck[ sTask ] ), uxPriority, ( xTaskHandle * ) NULL );
	}
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vCompeteingIntMathTask, pvParameters )
{
/* These variables are all effectively set to constants so they are volatile to
ensure the compiler does not just get rid of them. */
volatile portLONG lValue;
portSHORT sError = pdFALSE;
volatile signed portBASE_TYPE *pxTaskHasExecuted;

	/* Set a pointer to the variable we are going to set to true each
	iteration.  This is also a good test of the parameter passing mechanism
	within each port. */
	pxTaskHasExecuted = ( volatile signed portBASE_TYPE * ) pvParameters;

	/* Keep performing a calculation and checking the result against a constant. */
	for( ;; )
	{
	 /* Perform the calculation.  This will store partial value in
```

```
		registers, resulting in a good test of the context switch mechanism. */
		lValue = intgCONST1;
		lValue += intgCONST2;

		/* Yield in case cooperative scheduling is being used. */
		#if configUSE_PREEMPTION == 0
		{
			taskYIELD();
		}
		#endif

		/* Finish off the calculation. */
		lValue *= intgCONST3;
		lValue /= intgCONST4;

		/* If the calculation is found to be incorrect we stop setting the
		TaskHasExecuted variable so the check task can see an error has
		occurred. */
		if( lValue != intgEXPECTED_ANSWER ) /*lint !e774 volatile used to prevent this being
optimised out. */
		{
			sError = pdTRUE;
		}

		if( sError == pdFALSE )
		{
			/* We have not encountered any errors, so set the flag that show
			we are still executing.  This will be periodically cleared by
			the check task. */
			portENTER_CRITICAL();
				*pxTaskHasExecuted = pdTRUE;
			portEXIT_CRITICAL();
		}

		/* Yield in case cooperative scheduling is being used. */
		#if configUSE_PREEMPTION == 0
		{
			taskYIELD();
		}
		#endif
	}
}
/*-----------------------------------------------------------*/

/* This is called to check that all the created tasks are still running. */
portBASE_TYPE xAreIntegerMathsTaskStillRunning( void )
{
```

```
portBASE_TYPE xReturn = pdTRUE;
portSHORT sTask;

	/* Check the maths tasks are still running by ensuring their check variables
	are still being set to true. */
	for( sTask = 0; sTask < intgNUMBER_OF_TASKS; sTask++ )
	{
		if( xTaskCheck[ sTask ] == pdFALSE )
		{
			/* The check has not incremented so an error exists. */
			xReturn = pdFALSE;
		}

		/* Reset the check variable so we can tell if it has been set by
		the next time around. */
		xTaskCheck[ sTask ] = pdFALSE;
	}

	return xReturn;
}
```

pullQ.c

```
/*
	FreeRTOS.org V4.1.0 - Copyright (C) 2003-2006 Richard Barry.

	This file is part of the FreeRTOS.org distribution.
	****
*/

/*
 * This version of PollQ. c is for use on systems that have limited stack
 * space and no display facilities.  The complete version can be found in
 * the Demo/Common/Full directory.
 *
 * Creates two tasks that communicate over a single queue.  One task acts as a
 * producer, the other a consumer.
 *
 * The producer loops for three iteration, posting an incrementing number onto the
 * queue each cycle.  It then delays for a fixed period before doing exactly the
 * same again.
 *
 * The consumer loops emptying the queue.  Each item removed from the queue is
 * checked to ensure it contains the expected value.  When the queue is empty it
 * blocks for a fixed period, then does the same again.
 *
 * All queue access is performed without blocking.  The consumer completely empties
```

```
 * the queue each time it runs so the producer should never find the queue full.
 *
 * An error is flagged if the consumer obtains an unexpected value or the producer
 * find the queue is full.
 */

/*
Changes from V2.0.0

      + Delay periods are now specified using variables and constants of
        portTickType rather than unsigned portLONG.
*/

#include <stdlib.h>

/* Scheduler include files. */
#include "FreeRTOS.h"
#include "task.h"
#include "queue.h"

/* Demo program include files. */
#include "PollQ.h"

#define pollqSTACK_SIZE                     configMINIMAL_STACK_SIZE
#define pollqQUEUE_SIZE                     ( 10 )
#define pollqPRODUCER_DELAY          ( ( portTickType ) 200 / portTICK_RATE_MS )
#define pollqCONSUMER_DELAY          ( pollqPRODUCER_DELAY - ( portTickType ) 20 )
#define pollqNO_DELAY                ( ( portTickType ) 0 )
#define pollqVALUES_TO_PRODUCE       ( ( signed portBASE_TYPE ) 3 )
#define pollqINITIAL_VALUE           ( ( signed portBASE_TYPE ) 0 )

/* The task that posts the incrementing number onto the queue. */
static portTASK_FUNCTION_PROTO( vPolledQueueProducer, pvParameters );

/* The task that empties the queue. */
static portTASK_FUNCTION_PROTO( vPolledQueueConsumer, pvParameters );

/* Variables that are used to check that the tasks are still running with no
errors. */
static volatile signed portBASE_TYPE xPollingConsumerCount = pollqINITIAL_VALUE,
xPollingProducerCount = pollqINITIAL_VALUE;

/*-----------------------------------------------------------*/

void vStartPolledQueueTasks( unsigned portBASE_TYPE uxPriority )
{
static xQueueHandle xPolledQueue;
```

```
	/* Create the queue used by the producer and consumer. */
	xPolledQueue = xQueueCreate( pollqQUEUE_SIZE, ( unsigned portBASE_TYPE ) sizeof( unsigned portSHORT ) );

	/* Spawn the producer and consumer. */
	xTaskCreate( vPolledQueueConsumer, ( const signed portCHAR * const ) "QConsNB", pollqSTACK_SIZE, ( void * ) &xPolledQueue, uxPriority, ( xTaskHandle * ) NULL );
	xTaskCreate( vPolledQueueProducer, ( const signed portCHAR * const ) "QProdNB", pollqSTACK_SIZE, ( void * ) &xPolledQueue, uxPriority, ( xTaskHandle * ) NULL );
}
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vPolledQueueProducer, pvParameters )
{
unsigned portSHORT usValue = ( unsigned portSHORT ) 0;
signed portBASE_TYPE xError = pdFALSE, xLoop;

	for( ;; )
	{
	 for( xLoop = 0; xLoop < pollqVALUES_TO_PRODUCE; xLoop++ )
	 {
		/* Send an incrementing number on the queue without blocking. */
		if( xQueueSend( *( ( xQueueHandle * ) pvParameters ), ( void * ) &usValue, pollqNO_DELAY ) != pdPASS )
		{
			/* We should never find the queue full so if we get here there
			has been an error. */
			xError = pdTRUE;
		}
		else
		{
			if( xError == pdFALSE )
			{
				/* If an error has ever been recorded we stop incrementing the
				check variable. */
				portENTER_CRITICAL();
					xPollingProducerCount++;
				portEXIT_CRITICAL();
			}

			/* Update the value we are going to post next time around. */
			usValue++;
		}
	 }

	 /* Wait before we start posting again to ensure the consumer runs and
	 empties the queue. */
```

```
        vTaskDelay( pollqPRODUCER_DELAY );
       }
}  /*lint !e818 Function prototype must conform to API. */
/*-----------------------------------------------------------*/

static portTASK_FUNCTION( vPolledQueueConsumer, pvParameters )
{
unsigned portSHORT usData, usExpectedValue = ( unsigned portSHORT ) 0;
signed portBASE_TYPE xError = pdFALSE;

       for( ;; )
       {
        /* Loop until the queue is empty. */
        while( uxQueueMessagesWaiting( *( ( xQueueHandle * ) pvParameters ) ) )
        {
               if( xQueueReceive( *( ( xQueueHandle * ) pvParameters ), &usData, pollqNO_DELAY ) ==
pdPASS )
               {
                      if( usData != usExpectedValue )
                      {
                             /* This is not what we expected to receive so an error has
                             occurred. */
                             xError = pdTRUE;

                             /* Catch-up to the value we received so our next expected
                             value should again be correct. */
                             usExpectedValue = usData;
                      }
                      else
                      {
                             if( xError == pdFALSE )
                             {
                                    /* Only increment the check variable if no errors have
                                    occurred. */
                                    portENTER_CRITICAL();
                                           xPollingConsumerCount++;
                                    portEXIT_CRITICAL();
                             }
                      }

                      /* Next time round we would expect the number to be one higher. */
                      usExpectedValue++;
               }
        }

        /* Now the queue is empty we block, allowing the producer to place more
        items in the queue. */
```

```
		vTaskDelay( pollqCONSUMER_DELAY );
	}
} /*lint !e818 Function prototype must conform to API. */
/*-----------------------------------------------------------*/

/* This is called to check that all the created tasks are still running with no errors. */
portBASE_TYPE xArePollingQueuesStillRunning( void )
{
portBASE_TYPE xReturn;

	/* Check both the consumer and producer poll count to check they have both
	been changed since out last trip round.  We do not need a critical section
	around the check variables as this is called from a higher priority than
	the other tasks that access the same variables. */
	if( ( xPollingConsumerCount == pollqINITIAL_VALUE ) ||
		( xPollingProducerCount == pollqINITIAL_VALUE )
	  )
	{
		xReturn = pdFALSE;
	}
	else
	{
		xReturn = pdTRUE;
	}

	/* Set the check variables back down so we know if they have been
	incremented the next time around. */
	xPollingConsumerCount = pollqINITIAL_VALUE;
	xPollingProducerCount = pollqINITIAL_VALUE;

	return xReturn;
}
```