

## SYSTEM OPERACYJNY QNX 4.2x

System operacyjny QNX 4.2x jest systemem 32-bitowym opracowanym przez kanadyjską firmę QNX Software Systems, Ltd. należącym do rodziny systemów UNIX. System QNX 4.2x spełnia standardy POSIX 1003.1 wraz rozszerzeniami dotyczącymi systemów czasu rzeczywistego.

Jest to system operacyjny czasu rzeczywistego. Oznacza to, że oprócz podstawowego zadania każdego systemu operacyjnego, jakim jest zarządzanie zasobami komputera, ma on zdeterminowany maksymalny czas reakcji na zdarzenie zewnętrzne. Spełnienie tego warunku wymaga takiej konstrukcji systemu operacyjnego, aby zdeterminowane były :

- czas opóźnienia reakcji na przerwanie (ang.: *interrupt latency*) – czas liczony od wystąpienia przerwania do chwili rozpoczęcia wykonywania kodu funkcji obsługi danego przerwania oraz
- czas przełączania procesów (kontekstu) (ang.: *context switching*).

System QNX 4.2x jest systemem bardzo efektywnym w działaniu, mało obciążającym zasoby sprzętowe komputera oraz skalowalnym (to znaczy pozwalającym dobierać potrzebne komponenty systemu w zależności od potrzeb) w szerokim zakresie. Cechy te osiągnięto głównie dzięki architekturze system opartej na mikrojądrze.

Ciekawą cechą systemu QNX 4.2x jest zaimplementowanie protokołu sieciowego Native QNX Fleet pozwalający łączyć zasoby komputerów pracujących w sieci QNX w jeden logiczny komputer. Sieć ta zapewnia pełną „przezroczystość” w dostępie do zasobów niezależnie od tego, na którym węźle sieci się one znajdują.

## Charakterystyczne cechy systemu QNX 4.2x

System QNX 4.2x jako system czasu rzeczywistego posiada następujące cechy:

- Wielozadaniowość i wielowątkowość z priorytetami i wywłaszczaniem,
- Trzy algorytmy kolejkowania procesów:
  - FIFO (ang.: *First In, First Out*),
  - karuzelowy (ang.: *round-robin*)
  - adaptacyjny (ang.: *adaptive*)
- Synchronizacja procesów poprzez:
  - semafony,
  - synchroniczną wymianę wiadomości,
  - zastosowanie algorytmu kolejkowania FIFO (w ograniczonym zakresie)
- Komunikacja międzyprocesowa i międzywątkowa:
  - synchroniczna wymiana wiadomości (ang.: *message passing*),
  - pośtańcy (ang.: *proxy*),
  - sygnały,
  - pamięć dzielona (ang.: *shared memory*),
- Czasomierze,
- Zdeterminowany czas reakcji na przerwanie i czas przełączania kontekstu
- Protokół sieciowy Native QNX Fleet,
- Możliwość budowy sieci komputerowej wielokrotnej automatycznie balansowanej,
- Hierarchiczny system plików.
- Szerokie możliwości konfigurowania i skalowania.

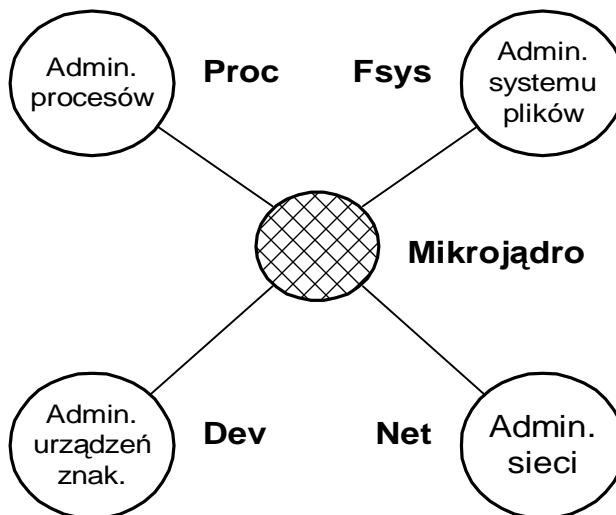
Wywłaszczanie oznacza, iż system operacyjny może przerwać wykonywanie procesu (wątku) (odebrać dostęp do procesora), przydzielając procesor innemu procesowi (wątkowi) na wykonywanie swoich zadań. Zatem system decyduje, jak długo dana aplikacja korzysta z procesora, i po upływie tego czasu następuje przekazanie kolejnej aplikacji procesorowi. (poprzednia zostaje wywłaszczona), lub jeśli dostępu do

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

procesora zażąda proces o wyższym priorytecie. Dzięki temu jeśli pojedynczy proces ulegnie zawieszeniu nie spowoduje zawieszenia całego systemu.

## Budowa systemu QNX 4.2x

W przeciwieństwie do innych systemów z rodziny systemów UNIX, których architektura bazuje na jądrze monolitycznym o rozmiarze około 1–2 MB, budowa systemu operacyjnego QNX 4.2x jest oparta na mikrojądrze, którego rozmiar wynosi około 15 kB kodu. Schematycznie przedstawiono to na rysunku 1.



Rys. 1. Ogólna budowa systemu QNX 4.2x

Mikrojądro jest odpowiedzialne za podstawowe funkcje systemowe oraz właściwą współpracę procesów i wątków. W systemie QNX 4.2x nie ma rozróżnienia pomiędzy procesami i wątkami systemowymi i użytkownika (za wyjątkiem można uznać systemowy administrator procesów, będący procesem o bardzo wysokim priorytecie). Schematyczną budowę mikrojądra systemu QNX 4.2x oraz powiązania między mikrojądrem i procesami pokazuje rysunek 2.

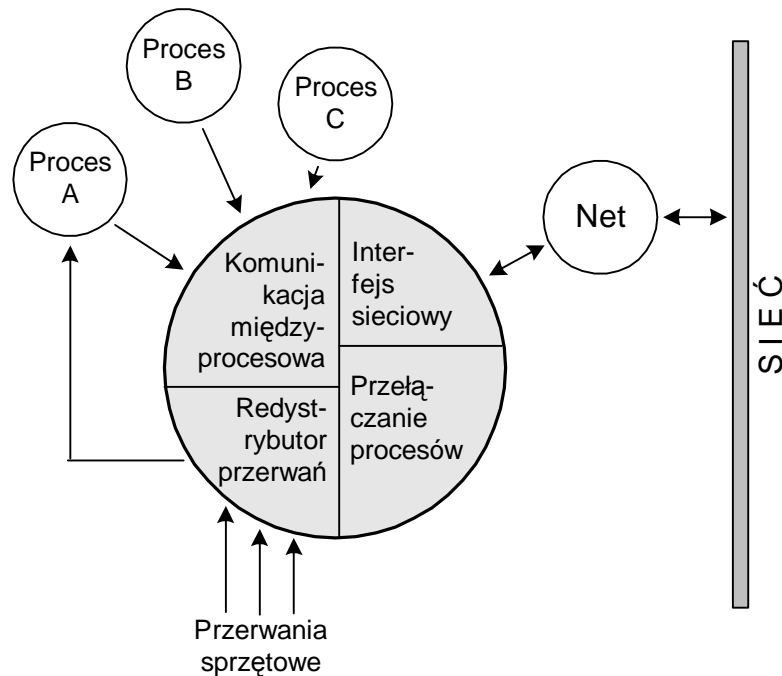
W mikrojądrze są zaimplementowane następujące funkcje systemowe:

- kolejkowanie i przełączenie procesów oraz wątków,
- komunikacja międzyprocesowa i międzywątkowa,
- niskopoziomowa obsługa przerw i ich redystrybucja,
- interfejs sieciowy pozwalający na wymianę wiadomości między procesami pracującymi na różnych komputerach połączonych siecią QNX Native Fleet.

Wszystkie pozostałe usługi systemu operacyjnego są zaimplementowane jako niezależne procesy systemowe nie różniące się od procesów użytkownika. Najczęściej wykorzystywanymi procesami systemowymi są:

- administrator procesów (*Proc*),
- administrator systemu plików (*Fsys*);
- administrator urządzeń znakowych (*Dev*),
- administrator sieci (*Net*).

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX



Rys. 2. Mikrojądro w systemie QNX 4.2x

## System plików

W systemie QNX 4.2x w systemie plików można wyróżnić następujące rodzaje plików:

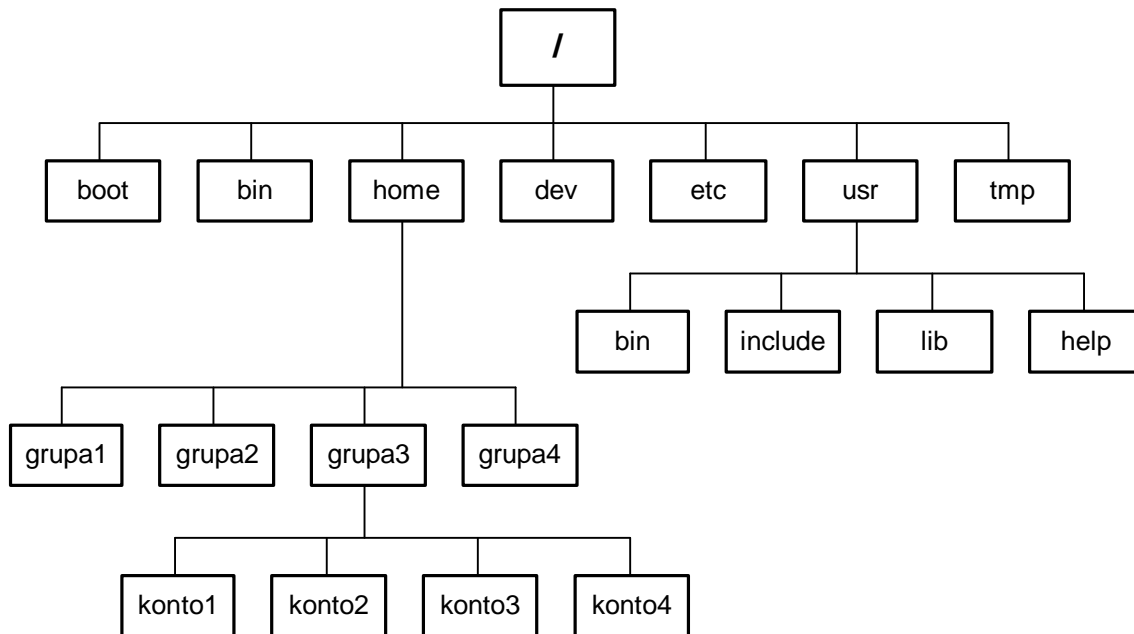
- Pliki normalne (ang.: *regular file*) – pliki nie posiadające określonej struktury służące do przechowywania danych,
- Katalogi (foldery) (ang.: *directories*) – pliki zawierające nazwy innych plików oraz informacje o ich miejscu na dysku, prawach dostępu, itp.
- Pliki specjalne – reprezentują urządzenia znakowe komputera (ang. *character special file*) i urządzenia blokowe (ang. *block special file*). Dzięki temu mechanizmowi są widzialne w systemie plików jako zbiory.
- Skróty (ang. *link*) – pliki zawierające nazwę ścieżki do innego pliku lub katalogu,
- Kolejki FIFO (ang. *fifo queue*), pliki będące kanałami wejścia–wyjścia,
- Potoki (ang.: *pipe*) – jednokierunkowe ograniczone bufony łączące standardowe wyjście jednego procesu ze standardowym wejściem innego.

Celem uporządkowania danych przechowywanych w systemie oraz ułatwienia dostępu do nich **system plików ma strukturę drzewiastą**. Podstawowy katalog zwany katalogiem głównym (ang.: *root directory*) jest oznaczany znakiem ukośnika „/”. W nim znajdują się pliki i inne katalogi, które z kolei mogą zawierać pliki i kolejne katalogi itd.

W systemie QNX w celu dostępu do danych znajdujących się na innym urządzeniu (np. napęd CD-ROM, napęd dyskietek) należy najpierw „zamontować” dane urządzenie w dowolnym miejscu systemu plików. Dzięki temu możliwy jest dostęp do danych bez konieczności znajomości urządzenia, na którym się one znajdują.

Struktura systemu plików w systemie QNX jest przedstawiona na rysunku 3.

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX



Rys. 3. Typowa struktura katalogów w systemie plików QNX

W poszczególnych katalogach pliki są pogrupowane następująco:

/	root, katalog główny,
bin	katalog zawierający zbiory wykonywalne systemu (polecenia),
boot	katalog zawierający narzędzia do wytwarzania obrazu systemu,
dev	pliki specjalne reprezentujące urządzenia fizyczne (dyski, konsole, terminale, napędy CD-ROM, porty szeregowy, porty równoległe itp.),
etc	katalog zawierające zbiory inicjujące i konfiguracyjne systemu,
usr	katalog zawierający programy użytkowe,
home	katalog zawierający konta użytkowników,
tmp	katalog plików tymczasowych,

Każdy plik ma swoją nazwę, która może się składać z dowolnej kombinacji cyfr, liter i innych znaków alfanumerycznych, z wyjątkiem znaków mających znaczenie specjalne, takich jak:

& > < ? ' " ` | / \

System rozróżnia duże i małe litery, na co należy zwracać szczególną uwagę, zwłaszcza przy przenoszeniu plików z systemu Windows.

**Dostęp do pliku** określa ścieżka dostępu czyli lista katalogów oddzielonych znakami „/”, przez które należy przejść celem dotarcia do określonego pliku. Rozróżnia się ścieżki względne i bezwzględne.

Nazwa ścieżki bezwzględnej zaczyna się od katalogu głównego danego (lokalnego) węzła (pojedynczy znak „/”) lub od podania numeru węzła (podwójny znak „//” i numer węzła). Podając ścieżkę bezwzględną identyfikacja danego pliku jest jednoznaczna i niezależna od tego, w którym katalogu w danej chwili użytkownik się znajduje.

Przykłady:

`/usr/lib/clib3r.lib` określa plik `clib3r.lib` w katalogu `/usr/lib` na węźle lokalnym  
`//3/bin/ls` określa plik `ls` w katalogu `/bin` na węźle numer 3

Ścieżka względna identyfikuje dany zbiór z punktu widzenia katalogu bieżącego (w którym aktualnie użytkownik pracuje). Przykładem najkrótszej możliwej ścieżki względnej jest podanie samej nazwy pliku lub katalogu. W takim przypadku jest to ścieżka z katalogu bieżącego do podanego pliku lub katalogu, który się w danym katalogu bieżącym znajduje.

Przykłady:

`abc.txt` określa plik `abc.txt` w katalogu bieżącym  
`../owady/mucha.c` określa plik `mucha.c` w katalogu `../owady` patrząc z katalogu bieżącego.

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

W systemie QNX dostęp do każdego pliku jest chroniony poprzez system **własności plików i praw dostępu**. Każdy plik ma swojego właściciela i jest przypisany do grupy użytkowników. Prawa dostępu podzielone są na trzy grupy: dostęp dla właściciela, grupy użytkowników i pozostałych. Możliwe do przydzielenia prawa to:

r      prawo czytania,  
w      prawo zapisu,  
x      prawo wykonywania pliku.

W przypadku katalogu prawo zapisu oznacza, iż zawarte w nim pliki można usuwać i zmieniać, natomiast prawo wykonywania pozwala na przeglądanie zawartości katalogu.

## PRACA W SYSTEMIE QNX

Każda osoba pracująca w systemie QNX ma swoje konto, czyli przydzieloną część obszaru w systemie plików. Autoryzacja dostępu do konta jest możliwa za pomocą hasła. Każdy użytkownik należy do określonej grupy (lub kilku grup).

Zakładając konto w systemie administrator określa m.in.:

- nazwę konta (login użytkownika)
- identyfikator (uid – user identification number),
- hasło,
- grupę (lub grupy) do jakich należy,
- katalog użytkownika (domowy),

Katalog użytkownika (katalog domowy) (ang.: *home directory*) jest to katalog zakładany podczas tworzenia konta każdemu użytkownikowi. Katalog ten należy tylko do tego danego użytkownika, mogącego tam umieszczać nowe pliki i tworzyć nowe katalogi. Pozostali użytkownicy (z wyjątkiem administratora) nie mają domyślnie pełnego dostępu do tego katalogu.

## Rozpoczęcie pracy z systemem

Użytkownicy rozpoczynający pracę w systemie rozpoznawani są za pomocą nazwy (ang.: *login*) i hasła (ang.: *password*). Jest wtedy otwierana konsola dla danego użytkownika. Użytkownik może otworzyć kilka konsol. Przełączanie między konsolami następuje poprzez naciśnięcie kombinacji klawiszy: Ctrl+Alt+n, gdzie n jest numerem konsoli tj. 1, 2, 3 itd. Dla przykładu chcąc przełączyć się na konsolę nr 2 należy nacisnąć kombinację klawiszy: Ctr+Alt+2

## Zakończenie sesji

W celu zakończenia sesji należy podać polecenie:

**logout**

lub nacisnąć klawisze Ctrl+D. Następuje wtedy zamknięcie danej konsoli. Chcąc wyłączyć komputer należy zamknąć system pisząc polecenie:

**shutdown** lub **shutdown -f**

## Wydawanie poleceń w systemie QNX

Komunikacja użytkownika z systemem QNX następuje poprzez wydawanie poleceń interpretowanych oraz realizowanych przez powłokę (ang.: *shell*). Składnia polecenia jest następująca:

**nazwa\_polec**    **opcje**    **argumenty [&]**

gdzie:

nazwa\_polec    komenda dostępna w systemie lub własny program użytkownika.  
opcje            jedna lub kilka opcji dostępnych z daną komendą. Opcje modyfikują i precyzują wykonanie danego polecenia. Najczęściej podaje się je jako pojedyncze litery po znaku „-”.  
argumenty      obiekty, na których wykonywane są polecenia, np. pliki, urządzenia zewnętrzne.

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

Zakończenie polecenia znakiem & powoduje uruchomienie procesu w tle. Oznacza to zwolnienie konsoli lub terminala przez dany proces i pozwala na pisanie nowych poleceń. W ten sposób są najczęściej uruchamiane sterowniki oraz usługi.

## PODSTAWOWE POLECENIA SYSTEMU QNX

### Znaki specjalne

Podczas pisania poleceń należy pamiętać o specjalnym znaczeniu następujących znaków:

- & przenoszenie procesu w tło („zerwanie” powiązania pomiędzy procesem a konsolą (terminalem), z którego proces uruchomiono)
- > przekierowanie wyniku polecenia (wyjścia),
- < przekierowanie pobierania danych do polecenia,
- | rozdzielenie kilku poleceń zapisanych w jednym wierszu w tzw. potok.
- \* oznacza wszystkie pliki.
- ? oznacza dowolną literę w nazwie pliku w miejscu, w którym znajduje się pytajnik

Wyżej opisanych znaków nie można używać w nazwach poleceń i nazwach plików.

### Operacje na plikach i katalogach.

#### ls

Wyświetla (listuje) zawartość katalogu. Składnia polecenia jest następująca:

```
ls [opcje] [nazwa]
```

Jeżeli nie jest podana nazwa katalogu lub zbioru, to wylistowany zostanie katalog bieżący.

Najczęściej stosowane opcje to:

- a wyświetlane są wszystkie zbiory (również te, których nazwa zaczyna się od znaku kropki).
- D wyświetlane są tylko katalogi
- l wyświetlanie pełnej informacji (długości, właściciela, grupie, prawach dostępu, datach utworzenia i modyfikacji) o poszczególnych zbiorach w danym katalogu.

Poszczególne opcje mogą być łączone ze sobą, np.

```
ls -al
```

wyświetla listę wszystkich plików w formacie pełnym.

#### pwd

Wyświetla bieżący katalog, w jakim użytkownik znajduje się w danym momencie.

#### cd

Zmienia katalog bieżący. Składnia polecenia jest następująca:

```
cd katalog
```

gdzie: *katalog* oznacza ścieżkę dostępu do nowego katalogu bieżącego.

Jako nazwa nowego katalogu mogą pojawić znaki mające specjalne znaczenie w systemie plików:

- cd / przejście do katalogu podstawowego,
- cd .. przejście o jeden katalog w górę (w hierarchii drzewa katalogów),
- cd ../.. przejście o dwa katalogi w górę,
- cd przejście do katalogu domowego z dowolnego miejsca.

Przykłady:

- cd mucha przejście do katalogu *mucha* znajdującego się w katalogu bieżącym
- cd /mucha przejście do katalogu *mucha* znajdującego się w katalogu głównym systemu plików
- cd /mucha/tse-tse przejście do katalogu *tse-tse* znajdującego w katalogu *mucha* znajdującego się w katalogu głównym systemu plików

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

## mkdir

Utworzenie nowego katalogu. Składnia polecenia jest następująca:

### **mkdir katalog**

gdzie: *katalog* określa nazwę katalogu, który ma być utworzony.

Przykłady:

`mkdir Trzmiel` tworzy katalog o nazwie `Trzmiel` w katalogu bieżącym

`mkdir ../osa` tworzy katalog o nazwie `osa` w katalogu o jeden poziom wyżej w stosunku do bieżącego katalogu.

## rmdir

Usunięcie katalogu. Usuwany katalog musi być pusty. Składnia polecenia jest następująca:

### **rmdir katalog**

## rm

Usunięcie zbioru(ów) lub drzewa katalogów. Składnia polecenia jest następująca:

### **rm [opcje] nazwa**

gdzie: *nazwa* określa zbiory lub katalogi które mają być usunięte.

Najczęściej używane opcje to:

- i system żąda potwierdzenia usunięcia każdego zbioru
- f system nie żąda potwierdzenia usunięcia zbiorów, które mają zabroniony zapis
- v w czasie usuwania są listowane usuwane zbiory
- R usuwanie całego drzewa katalogów łącznie z podkatalogami
- d usuwanie plików z całego drzewa katalogów bez usuwania samej struktury katalogów

Przykłady:

`rm MojZbior` usuwa zbiór o nazwie `MojZbior` z bieżącego katalogu.

`rm -v katalog/*` usuwa wszystkie zbiory w katalogu *katalog* i listuje usuwane zbiory.

`rm -vR *` usuwa wszystkie zbiory i katalogi z katalogu bieżącego i listuje usuwane zbiory.

## mv

Przenoszenie lub zmiana nazwy zbiorów. Składnia polecenia jest następująca:

### **mv [opcje] plik1 plik2**

gdzie: *plik1* określa zbiór, który ma być przeniesiony do miejsca określonego przez nazwę *plik2*. Jeśli nazwa *plik2* identyfikuje zbiór (nie katalog) w tym samym katalogu to następuje zmiana nazwy pliku, w przeciwnym wypadku plik jest przeniesiony do katalogu określonego nazwą *plik2*. Jeżeli ma być przeniesionych więcej niż jeden plik (np. poprzez użycie znaków `*` lub `?` w nazwie *plik1*), to nazwa *plik2* musi określać katalog.

Na przykład:

`mv jeden.txt dwa.txt` zmienia nazwę pliku `jeden.txt` na `dwa.txt`.

`mv jeden.txt abc` jeżeli `abc` jest nazwą katalogu w katalogu bieżącym to plik `jeden.txt` zostanie przeniesiony do katalogu `abc` bez zmiany nazwy. W katalogu bieżącym plik `jeden.txt` przestaje istnieć.

## cp

Kopiowanie zbioru(ów) lub drzewa katalogów. Składnia polecenia jest następująca:

### **cp[opcje] plik1 plik2**

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

gdzie: `plik2` określa nazwę zbioru lub katalogu docelowego, do którego ma być skopiowany `plik1`. Jeżeli ma być skopiowanych więcej niż jeden plik (np. poprzez użycie znaków `*` lub `?` w nazwie), to nazwa `plik2` musi określać katalog.

Najczęściej używane opcje to:

- `-v` w trakcie kopiowania są listowane zbiory kopiowane.
- `-R` kopiowanie drzewa katalogów. Parametr ten musi być użyty jeśli zbiorem źródłowym jest katalog.
- `-i` podczas kopiowania system żąda potwierdzenia, jeżeli nastąpi zapisanie kopiowanego zbioru na już istniejącym.
- `-c` w czasie kopiowania tworzone jest automatycznie drzewo katalogów, takie jak w katalogu źródłowym. Używana jest przeważnie łącznie z opcją `-R`

Przykłady:

<code>cp jeden.txt dwa.txt</code>	kopiuje zbiór <code>jeden.txt</code> na zbiór <code>dwa.txt</code> .
<code>cp -v oko* kopia/</code>	kopiuje wszystkie zbiory, których nazwy zaczynają się ciągiem znaków 'oko' do katalogu <code>kopia</code> i listuje nazwy zbiorów kopiowanych.
<code>cp -vRc * ../backup/</code>	kopiuje wszystkie zbiory i katalogi z katalogu bieżącego do katalogu <code>backup</code> znajdującego się o jeden poziom wyżej w stosunku do bieżącego i listuje nazwy kopiowanych zbiorów.

## ln

Tworzenie linku (dowiązania) do pliku. Nazwa będąca dowiązaniem do pliku może znajdować się w zupełnie innym katalogu niż plik macierzysty.

Rozróżnia się linki twarde (ang.: *hard link*) i linki symboliczne (ang.: *symbolic link*). Link twarty to nadanie innej nazwy istniejącemu plikowi (przez to link oraz oryginał stają się przez to nierozróżnialne). Link symboliczny jest specjalnym typem pliku wskazującym na plik oryginalny poprzez jego nazwę (jest to wpis w katalogu wskazującym na nazwę oryginalnego pliku).

Składnia polecenia `ln`, w przypadku tworzenia linku twardego o nazwie `plik2` do pliku o nazwie `plik1` wygląda następująco:

```
ln plik1 plik2
```

Dodanie opcji `-s` pozwala na utworzenie linku symbolicznego.

## find

Wyszukiwarka systemowa. Najczęściej wykorzystuje się ją do znajdowania na dysku pliku o nazwie `plik` rozpoczynając szukanie od katalogu `katalog`. W tym przypadku format polecenia jest następujący:

```
find katalog -name plik
```

## cat

Czyta kolejno wyszczególnione pliki `plik1`, `plik2`, itd. i wypisuje ich zawartość do standardowego wyjścia (stdout).

```
cat plik1 plik2 plik3
```

## more

Filtr umożliwiający wyświetlanie zawartości pliku tekstowego o nazwie `plik` na ekranie konsoli (terminala) w sposób umożliwiający wstrzymanie wyświetlania po każdym zapełnieniu ekranu.

```
more plik
```

## less

Podobnie jak `more` z tym, że umożliwia także przeglądanie wstecz.

```
less plik
```

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

## WC

Zliczanie liczby linii, słów lub znaków w pliku o nazwie `plik`. Składnia polecenia jest następująca:

**wc [opcje] plik**

Znaczenie opcji jest następujące:

- c zliczanie znaków,
- l zliczanie linii,
- w zliczanie słów.

## chmod

Służy do zmiany uprawnień użytkowników w stosunku do pliku czy katalogu. Składnia polecenia jest następująca:

**chmod prawa\_dostępu plik**

Prawa dostępu można podawać w dwojaki sposób:

- w postaci liczbowej z zastosowaniem zapisu ósemkowego, przy założeniu, że prawu odczytu `r` odpowiada wartość 4, prawu zapisu `w` – 2, a prawu wykonywania `x` – 1, brak danego prawa równoważny jest 0. W ramach danej grupy użytkowników wartości liczbowe sumują się.
- w postaci specyfikacji: `u[+|-|=][rwx]`, `g[+|-|=][rwx]`, `o[+|-|=][rwx]`. Litery 'u', 'g' i 'o' oznaczają prawa właściciela, grupy i innych użytkowników. Znak '+' oznacza dodanie prawa, znak '-' powoduje odebranie prawa, a znak '=' – ustawienie prawa. Litery 'r', 'w' i 'x' oznaczają prawo odczytu, zapisu i wykonania.

Przykładowo:

```
chmod u=rwx,g=rx,o=r mojplik
```

```
chmod 754 mojplik
```

ustawiają te same prawa dostępu, czyli właściciel pliku `mojplik` będzie miał wszystkie uprawnienia (`rwX`, czyli liczbowo  $4+2+1=7$ , stąd pierwsza liczba 7 dla użytkownika), członkowie grupy będą mogli wykonać go lub odczytać (`r-x`, czyli  $4+0+1=5$ ), a inni mogą go tylko odczytać.

## chown

Służy do zmiany właściciela pliku i ewentualnie grupy, do której dany plik należy. Składnia polecenia jest następująca:

**chown [-R] właściciel[:grupa] plik**

gdzie: `właściciel` określa nowego właściciela pliku `plik`. Opcjonalnie można zmienić grupę związaną z danym plikiem podając po dwukropku jej nazwę.

Najczęściej używane opcje to:

- R jeżeli nazwa plik jest katalogiem, to zmiana właściciela i grupy następuje dla danego katalogu i rekursywnie dla wszystkich plików i katalogów w nim występujących.

## mount

Służy do „montowania” (ustalania miejsca) urządzenia blokowego w systemie plików. Składnia polecenia jest następująca:

**mount nazwa\_urządzenia katalog**

gdzie: `nazwa_urządzenia` jest urządzeniem (znajdującym się w katalogu systemowym `/dev`), które ma być zamontowane. `katalog` określa miejsce zamontowania w systemie plików.

Polecenie to jest najczęściej używane montowania napędu dyskietek (z dyskietką sformatowaną pod systemem QNX) w systemie plików.

Przykładowo:

```
mount /dev/fd0 /fd0
```

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

montuje napęd dyskietek (`/dev/fd0`) w katalogu głównym systemu plików pod nazwą `fd0`. Po zamontowaniu dostęp do plików na dyskietce jest możliwy poprzez napisanie ścieżki dostępu `/fd0/nawa_pliku`.

```
mount /dev/fd0 fd
```

montuje napęd dyskietek (`/dev/fd0`) w katalogu bieżącym systemu plików pod nazwą `fd`.

## umount

Służy do zamknięcia systemu plików na danym urządzeniu i „odmontowania” urządzenia. Składnia polecenia jest następująca:

### **umount katalog**

gdzie: `katalog` określa miejsce w systemie plików urządzenia, które ma być odmontowane.

Odmontowanie dyskietki jest ważne przed jej wyjęciem, szczególnie wtedy, gdy były na nią zapisywane pliki.

## Fatfsys

Uruchomienie sterownika umożliwiającego dostęp do systemu plików FAT12, FAT16 i FAT32. Pozwala ono na odczytywanie i zapisywanie dyskietek i partycji dysków obsługiwanych przez system Windows (z wyjątkiem systemu plików NTFS).

Najczęściej sterownik ten uruchamia się pisząc polecenie:

### **Fatfsys &**

Powoduje ono uruchomienie sterownika i automatycznie montuje w systemie plików napęd dyskietek oraz partycje dysków sformatowane w systemie FAT jako `/dos/a`, `/dos/c` itd.

## **Inne polecenia**

### passwd

Umożliwia zmianę hasła wpisywanego przez użytkownika przy logowaniu się do systemu. Na żądanie systemu należy podać kolejno stare hasło i dwukrotnie nowe.

### logout

Zamknięcie konsoli lub terminala. Jest równoważne naciśnięciu kombinacji klawiszy `Ctrl+D`.

### date

Aktualna data i godzina.

### df

Wyświetla informacje o wolnym miejscu na dysku. Miejsce dysku w systemie plików określa `plik`. Składnia polecenia jest następująca:

```
df [opcje] [-n numer] plik
```

Najczęściej używane opcje to:

- `-h` wyświetlany jest nagłówek w informacji o wolnej przestrzeni na dysku,
- `-n numer` wyświetla informację o wolnym miejscu na dysku na węźle `numer`,
- `-b` informacje są wyświetlane jako liczba 512-bajtowych bloków (nie w KB).

### du

Wyświetla informacje o zajętości dysku `plik` lub katalog `plik`. Składnia polecenia jest następująca:

```
du [-a|-s] [-p] plik
```

Najczęściej używane opcje to:

- `-a` wyświetlana jest informacja dla każdego pliku oddzielnie,
- `-s` wyświetlana jest informacja łączna dla całego katalogu,

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

- p zajętość jest wyświetlana w bajtach (domyślnie jako liczba 512-bajtowych bloków)
- k zajętość jest wyświetlana w kilobajtach (domyślnie jako liczba 512-bajtowych bloków)

## use

Wyświetla tekst pomocy dla danego polecenia. Składnia polecenia `use` jest następująca:

**use nazwa\_polecenia**

gdzie: `nazwa_polecenia` jest nazwą polecenia systemowego.

## **Polecenia specyficzne dla systemu QNX**

### alive

Wyświetla status wszystkich węzłów w sieci QNX. Status poszczególnych węzłów jest przedstawiany jako „Up” (węzeł pracujący, widoczny) albo „Down” (węzeł niepracujący, uszkodzony itp).

### sin

Wyświetla informację o stanie systemu. Polecenie to ma duże znaczenie diagnostyczne, szczególnie przy rozwiązywaniu problemów z systemem. Składnia polecenia `sin` jest następująca:

**sin [opcje] [polecenie]**

Najczęściej używanymi opcjami są:

- n *numer* powoduje wyświetlenie informacji o systemie zainstalowanym na węźle o numerze *numer* (domyślnie polecenie dotyczy węzła lokalnego),
- P *nazwa* dostarcza informacji o procesach o podanej nazwie,
- p *pid* dostarcza informacji o procesie o danym numerze *pid*.

Polecenie specyfikuje rodzaj informacji, jakie dostarczane są przez narzędzie `sin`. Wyszczególnienie polecenia jest opcjonalne. Jego brak powoduje wyświetlenie informacji o procesach uruchomionych na danym węźle:

- ścieżkę dostępu do programu (PROGRAM),
- numer identyfikacyjny procesu (PID),
- numer sesji, z której uruchomiono program (SID),
- priorytet i algorytm kolejkowania (PRI),
- stan procesu i numer identyfikacyjny procesu blokującego (STATE i BLK)
- zajętość pamięci przez segment kodu i danych (CODE i DATA).

Najczęściej używanymi poleceniami są:

- `args` powoduje wyświetlenie argumentów, a jakimi program został uruchomiony w linii poleceń,
- `env` powoduje wyświetlenie zmiennych środowiskowych dostępnych dla danego programu,
- `times` powoduje wyświetlenie czasu procesora wykorzystanego przez dany proces. Wyświetlany jest:
  - czas uruchomienia procesu (START TIME),
  - czas systemowy wykorzystany przez proces (STIME),
  - czas użytkownika wykorzystany przez proces (UTIME),
  - czas systemowy i czas użytkownika wykorzystane przez procesy potomne (ang.: *child process*) (CSTIME i CUTIME).Pozostałe informacje są identyczne z wyświetlanymi przez `sin` bez podawania polecenia (PROGRAM, PID, SID i PRI)
- `names` powoduje wyświetlenie nazw zarejestrowanych na danym węźle.
- `gnames` powoduje wyświetlenie globalnych nazw zarejestrowanych na wszystkich węzłach i węzłach, na którym dana nazwa jest zarejestrowana.
- `proxies` listuje posłańców (ang.: *proxies*) wysłanych do danego procesu. Wyświetlany jest:
  - numer identyfikacyjny (PID),
  - proces będący właścicielem posłańca (PROGRAM),
  - priorytet i stan posłańca (PRI i STATE),
  - liczba posłańców oczekujących na odebranie (COUNT).
- `info` wyświetlane są ogólne informacje o systemie (rodzaj procesora, wielkość pamięci itp.).

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

Przykładowo:

```
sin
sin -n 3
sin times
sin -n 2 -P Net
```

## slay

Polecenie `slay` pozwala usunąć (zabić) proces poprzez podanie jego nazwy (a nie numeru identyfikacyjnego `pid`, jak w przypadku polecenia `kill`). Składnia polecenia `slay` jest następująca:

**slay [opcje] nazwa\_programu**

Najczęściej używanymi opcjami są:

- `-n numer` powoduje zabicie procesu na węźle o numerze `numer` (domyślnie polecenie dotyczy węzła lokalnego),
- `-f` wymusza zabicie wszystkich procesów o danej nazwie. Gdy opcja ta nie jest podana, `slay` pyta użytkownika o zgodę na zabicie danej instancji procesu podając jej `pid`.

## netinfo

Polecenie `netinfo` wyświetla zdarzenia, jakie zostały zarejestrowane przez administratora sieci lub informacje diagnostyczne związane z interfejsem sieciowym. Informacje te są dostarczane przez administrator sieci `Net` albo sterownik karty sieciowej. Składnia polecenia `netinfo` jest następująca:

**netinfo [-n numer] [-l]**

Najczęściej używanymi opcjami są:

- `-n numer` powoduje wyświetlenie informacji o pracy sieci na węźle o numerze `numer` (domyślnie polecenie dotyczy węzła lokalnego),
- `-l` wyświetla informacje diagnostyczne interfejsu sieciowego.

Polecenie `netinfo` dostarcza bardzo dużo danych na temat pracy sieci (ilości pakietów wysłanych, odebranych, ilości kolizji itp.). Jest ono bardzo użyteczne przy rozwiązywaniu problemów z siecią.

## on

Uruchamia program (inne polecenie) na zdalnym węźle sieci QNX. Format polecenia `on` jest następujący:

**on [opcje] polecenie**

gdzie `polecenie` jest programem, który ma być uruchomiony na zdalnym węźle. Najczęściej używane opcje to:

- `-n numer` powoduje uruchomienie polecenia na węźle `numer`,
- `-f numer` powoduje uruchomienie polecenia na węźle `numer` i ustala dla uruchamianego polecenia katalog główny danego węzła, jako `root` dla całej sieci.

Przykładowo, jeżeli założyć, że poniższe polecenia są wydawane na węźle 1, to:

```
on -n 5 ls /
```

powoduje uruchomienie polecenia `ls` na węźle 5, ale wylistowany będzie katalog katalog węzła 1, ponieważ symbol `./` będzie interpretowany jako katalog `root` węzła 1. Natomiast polecenie

```
on -f 5 ls /
```

powoduje uruchomienie polecenia `ls` na węźle 5, ale wylistowany będzie katalog katalog węzła 5, ponieważ symbol `./` będzie w tym przypadku interpretowany jako katalog `root` węzła 5.

## ditto

Zdalna konsola. Polecenie to pozwala skojarzyć lokalną konsolę z inną (najczęściej pracującą na zdalnym węźle). Format polecenia jest następujący:

**ditto ścieżka\_do\_konsoli**

gdzie `ścieżka_do_konsoli` oznacza ścieżkę do konsoli (najczęściej zdalnej), z którą chcemy się połączyć.

# Wprowadzenie do systemu operacyjnego czasu rzeczywistego QNX

Przykładowo, jeśli wydamy polecenia:

```
ditto //3/dev/con2
```

to lokalna konsola zostanie skojarzona z konsolą 2 (urządzenie `dev/con2`) na węźle 3. Od tego momentu można pracować na lokalnej konsoli tak jakby było się zalogowanym na węźle 3 i pracowało na konsoli 2.

W celu wydania poleceń sterujących lokalnej konsoli należy nacisnąć klawisze `Ctrl-E` oraz polecenie.

Najczęściej wykorzystywanymi poleceniami są:

`Ctrl-E k` powoduje załączenie albo wyłączenie lokalnej klawiatury jako zdalnej,

`Ctrl-E q` wyjście z programu ditto.

## ***Niektóre programy użytkowe***

### vedit

Pełnoekranowy edytor tekstu. Pozwala na edycję wielu plików jednocześnie w kilku oknach. Korzystanie z edytora vedit jest intuicyjne i łatwe dzięki dostępowi do funkcji edytora z menu lub za pomocą skrótów klawiaturowych.

### cc

Kompilator systemowy. Polecenie to uruchamia kompilator odpowiedniego języka (C, C++, asemblera, itd.) i ewentualnie linker. Składnia polecenia jest następująca:

```
cc [opcje] plik
```

gdzie: `plik` jest nazwą zbioru w danym języku, który ma zostać poddany kompilacji.

W zależności od rozszerzenia nazwy pliku jest uruchamiany kompilator odpowiedniego języka.

Przykłady:

```
cc program.c
```

następuje kompilacja i konsolidacja programu o nazwie `program.c` z zapisaniem kodu wynikowego w pliku `a.out`. Do kompilacji zostanie użyty kompilator języka C (`wcc386`) i linker (`wlink`).

```
cc -o abc prog.cpp
```

następuje kompilacja i konsolidacja programu o nazwie `prog.cpp` z zapisaniem kodu wynikowego w pliku `abc`. Do kompilacji zostanie użyty kompilator języka C++ (`wpp386`) i linker (`wlink`).

### mc

Uruchomienie Midnight Commandera – tekstowego menedżera plików mogącego pracować pod konsolą lub terminalem. Narzędzie to jest także dostępne w innych systemach z rodziny systemu Unix (np. w systemie Linux).

### ph

Uruchomienie środowiska Photon – graficznej powłoki systemu QNX.