Systemy czasu rzeczywistego OS/2 i QNX

- *System operacyjny czasu rzeczywistego*
  - *Szeregowanie zadań i podział czasu*
  - *Wywłaszczenia*
- *OS/2*
  - *Historia*
  - *Zalety*
- *QNX*
  - *Architektura*
  - *Komunikacja*
  - *Przerwania i czas*
  - *Sieć*

System operacyjny czasu rzeczywistego

System operacyjny czasu rzeczywistego pozwala między innymi na szeregowanie zadań oraz podział czasu procesora w celu jednoczesnej realizacji wielu zadań.

Zastosowania: technika wojskowa, automatyka przemysłowa, robotyka, technika motoryzacyjna.

System operacyjny czasu rzeczywistego

Szeregowanie zadań i podział czasu

Jedno z najważniejszych zadań systemu czasu rzeczywistego polega na podziale czasu procesora i praw dostępu do zasobów systemowych pomiędzy konkurujące ze sobą zadania.

Algorytm szeregowania zadań polega na przydzielaniu zadań procesorowi (lub procesorom) w taki sposób, aby każde zadanie było zrealizowane w całości.

Algorytm podziału czasu procesora (procesorów) dokonuje podziału dysponowanego czasu na przedziały, przyporządkowując każdemu z nich numer procesu, który ma się w tym przedziale wykonywać.

Szeregowanie zadań i podział czasu pozwalają na wykonanie wielu procesów jednocześnie. Za ustalenie kolejności zadań odpowiedzialny jest planista. Poprawne funkcjonowanie algorytmu szeregowania zadań wymaga zastosowania mechanizmu wywłaszczeń.

System operacyjny czasu rzeczywistego

Wywłaszczenia

Wywłaszczenie jest narzędziem planisty (algorytmu szeregującego) polegającym na wstrzymaniu jednego procesu i umożliwieniu wykonania innego procesu. W ten sposób wyklucza się możliwość dominacji niektórych procesów.

Jednak nie wszystkie procesy mogą być wywłaszczane. Do procesów takich należą te, które dotyczą np. obsługi przerwań systemowych lub sprzętowych oraz niektórych zadań zadań krytycznych czasowo. Racjonalnie, procesy takie powinny mieć zapewniony zdeterminowany i możliwie krótki czas realizacji.

Systemy nie posiadające mechanizmu wywłaszczania:

Windows 3.x

Posiadające mechanizm wywłaszczania:

Systemy linuxowe > 2.6

Windows 95, nt, me, 2k, xp, vista

System operacyjny OS/2

OS/2 - historia

1987 – wersja 1.0 IBM i Microsoft. Oceniany pozytywnie API (*Application Programming Interface*) na poziomie procedur i funkcji oraz komunikacji z bibliotekami i systemem operacyjnym. API stworzony był z myślą o uproszczeniu tworzenia aplikacji.

API: np. Win32 (kernel32.dll) – 32 bitowa baza programisty

DirectX – zestaw funkcji wspomagający operacje graficzne

1990 – sukces systemu Windows, podział IBM OS/2 2.0 i Microsoft NT OS/2 3.0 -> Windows NT (*New Technology*) nowa rodzina systemów operacyjnych

Windows NT 3.1
Windows NT 5.0 (Windows 200)
Windows NT 5.1 (Windows XP)
Windows NT 6.0 (Windows Vista)

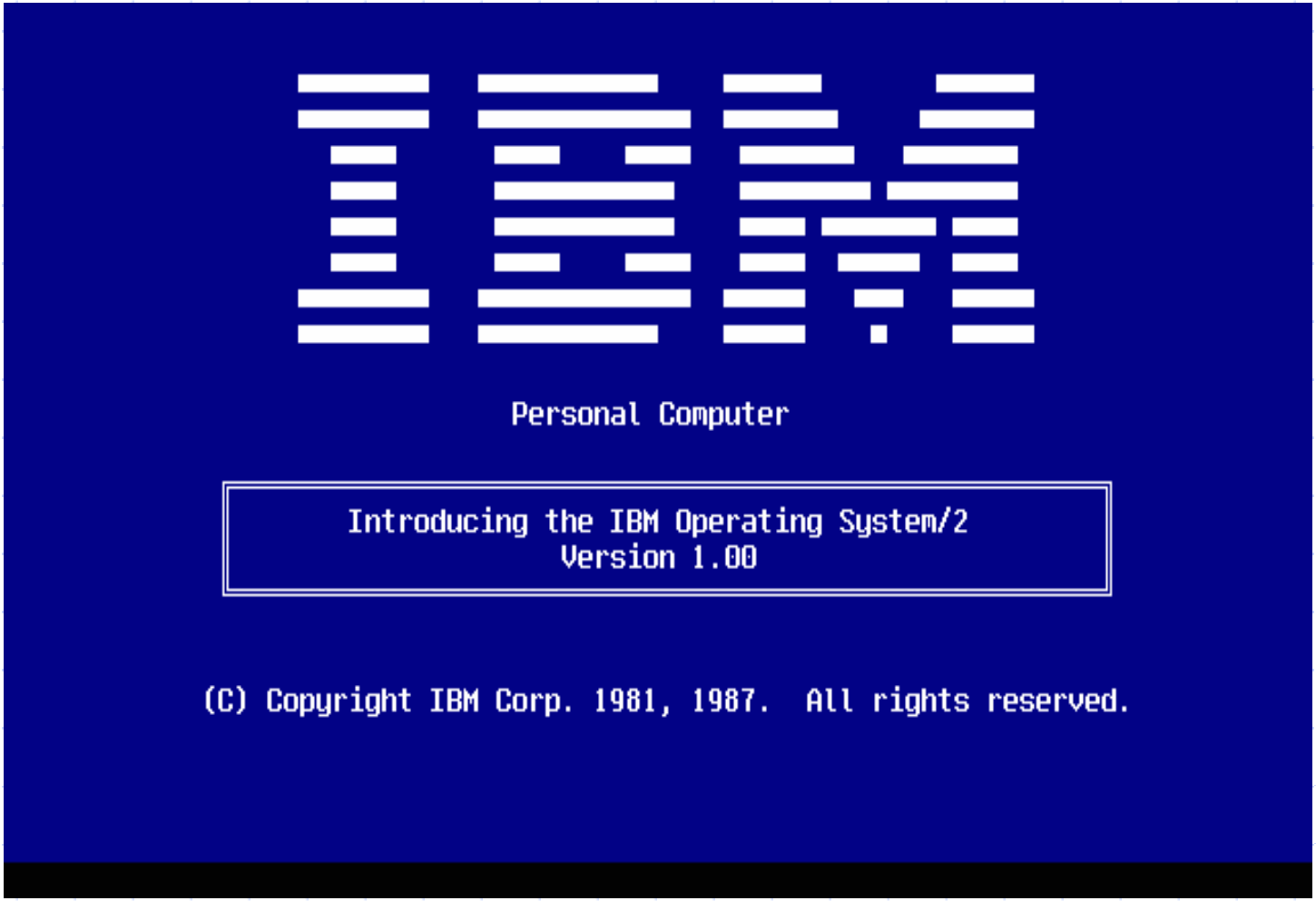
1995 - IBM wycofuje się z prac nad OS/2

2005 – oficjalne wycofanie łatek i wszelkiego wsparcia dla systemu

System operacyjny OS/2

OS/2 - historia

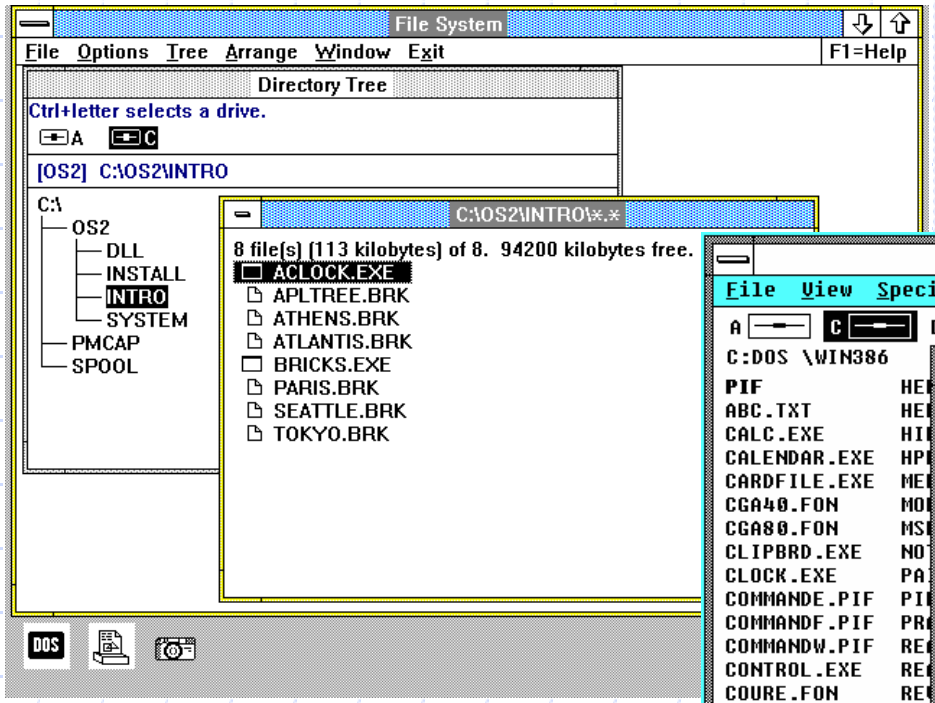
Pierwsza wersja bez interfejsu graficznego



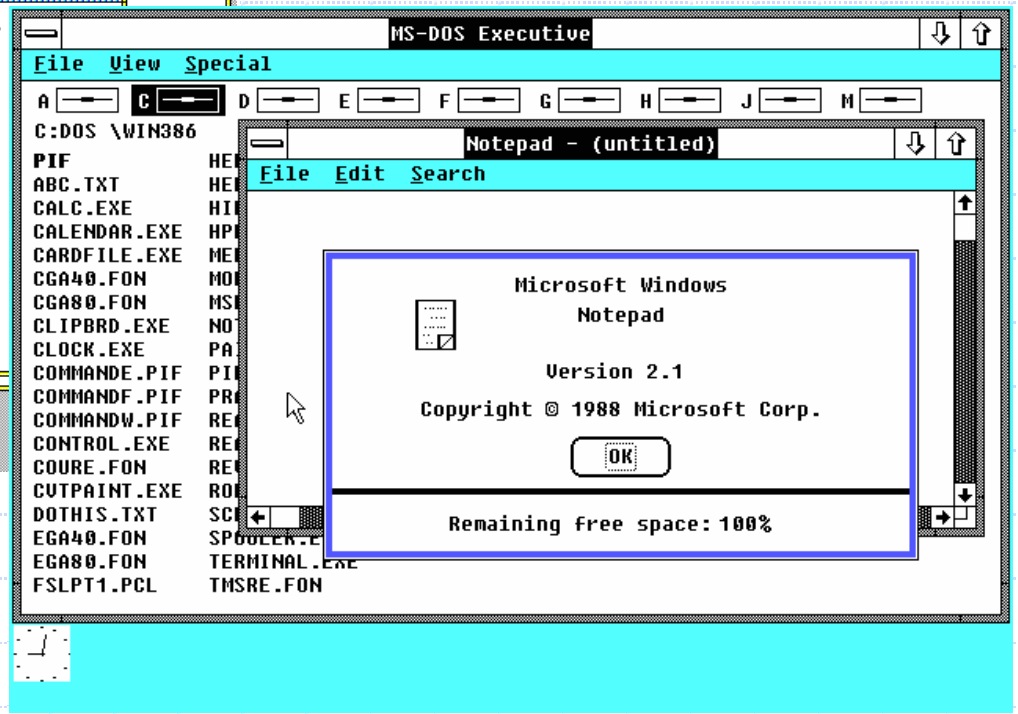
System operacyjny OS/2

OS/2 - historia

Interfejs graficzny OS/2 1.1



Interfejs graficzny Windows 2.1



System operacyjny OS/2

OS/2 - zalety

- Wielozadaniowość. Możliwość jednoczesnego uruchomienia wielu zadań. Żadne z zadań nie może absorbować pełnych zasobów CPU.
- Wielowątkowość – wykonywanie zadań bez konieczności oczekiwania na zakończenie poprzedniego zadania (formatowanie, drukowanie, edytowanie, archiwizowanie – do 512 wątków w OS/2 1.3).
- Wzbogacone możliwości konfiguracyjne przez możliwość doboru własnych sterowników dla określonej aplikacji.
- Ochrona przed zawieszeniem systemu. Każda aplikacja jest izolowana od drugiej, co pozwala na ochronę obszarów pamięci.
- Możliwość wykorzystania interfejsu graficznego użytkownika WorkShell.
- Dostęp do wyspecjalizowanych aplikacji pracujących jedynie pod OS/2.

System operacyjny QNX

QNX – następca OS/2 ?

- Unixowy system opracowany przez Quantum Software na komputery IBM x86.
- Napisany i obsługiwany w C.
- Zgodny z normą POSFIX (Portable Operating System Interface)
- Standaryzuje: interfejs programistyczny (API) i interfejs użytkownika.
- Właściwości powłoki: jest zgodny z większością systemów: Linux, Unix, BSD.
- Elastyczna konfiguracja każdego modułu systemu (zadania systemowego).
- Dynamiczna zmiana konfiguracji

System operacyjny QNX

QNX – architektura

Egzekutor jądra systemu szereguje zadania, implementuje komunikację pomiędzy zadaniami, przyjmuje przerwania.

Zadania systemowe:

1. Administrator zadań – tworzenie i usuwanie zadań, odmierzanie czasu.
2. Administrator sieci – tworzenie łącz sieciowych i organizacja komunikacji.
3. Administrator zbiorów – system zarządzania zbiorami pamięci
4. Administrator urządzeń – kanał: zadania<->urządzenia zewnętrzne
5. Zadania sterujące urządzeniami – sterowanie przerwaniami

Komunikacja zadań jest realizowana przez jądro systemu. Występuje hierarchia powiązań. Procesy macierzyste mają kontrolę nad procesami potomnymi. Istnieje możliwość zawieszenia i zerwania procesu. Funkcja *wait* zawiesza wykonanie procesu macierzystego do chwili wykonania procesu potomnego. Zakończenie procesu macierzystego kończy wszystkie jego procesy potomne.

System operacyjny QNX

QNX – architektura

Egzekutor jądra wyróżnia 32 priorytety zadań. Algorytmy szeregowania zadań:

- Algorytm FIFO. Zadanie najdłużej oczekujące w kolejce jest wykonywanie w pierwszej kolejności aż do zakończenia lub wywłaszczenia przez siebie lub inne zadanie. Pozwala to na zmniejszenie czasów przełączania zadań.
- Algorytm karuzelowy (planowanie rotacyjne) polega na tym, że każde z zadań dostaje określony przedział (kwant) czasu. Po wyczerpaniu tego czasu, zadanie zostaje wywłaszczone i ustawione na końcu kolejki.
- Algorytm adaptacyjny – jest podobny do algorytmu karuzelowego z tym, że niewykonanie zadania w przydzielonym kwancie czasu powoduje obniżenie jego priorytetu o 1. Natomiast nie podjęte w czasie 1 s zadanie aktywne zwiększa priorytet o 1. Algorytm ten faworyzuje krótkie zadania często wykonywane i dyskryminuje zadania czasochłonne np. do obsługi aplikacji wieloportalowych.

System operacyjny QNX

QNX – komunikacja - spotkania

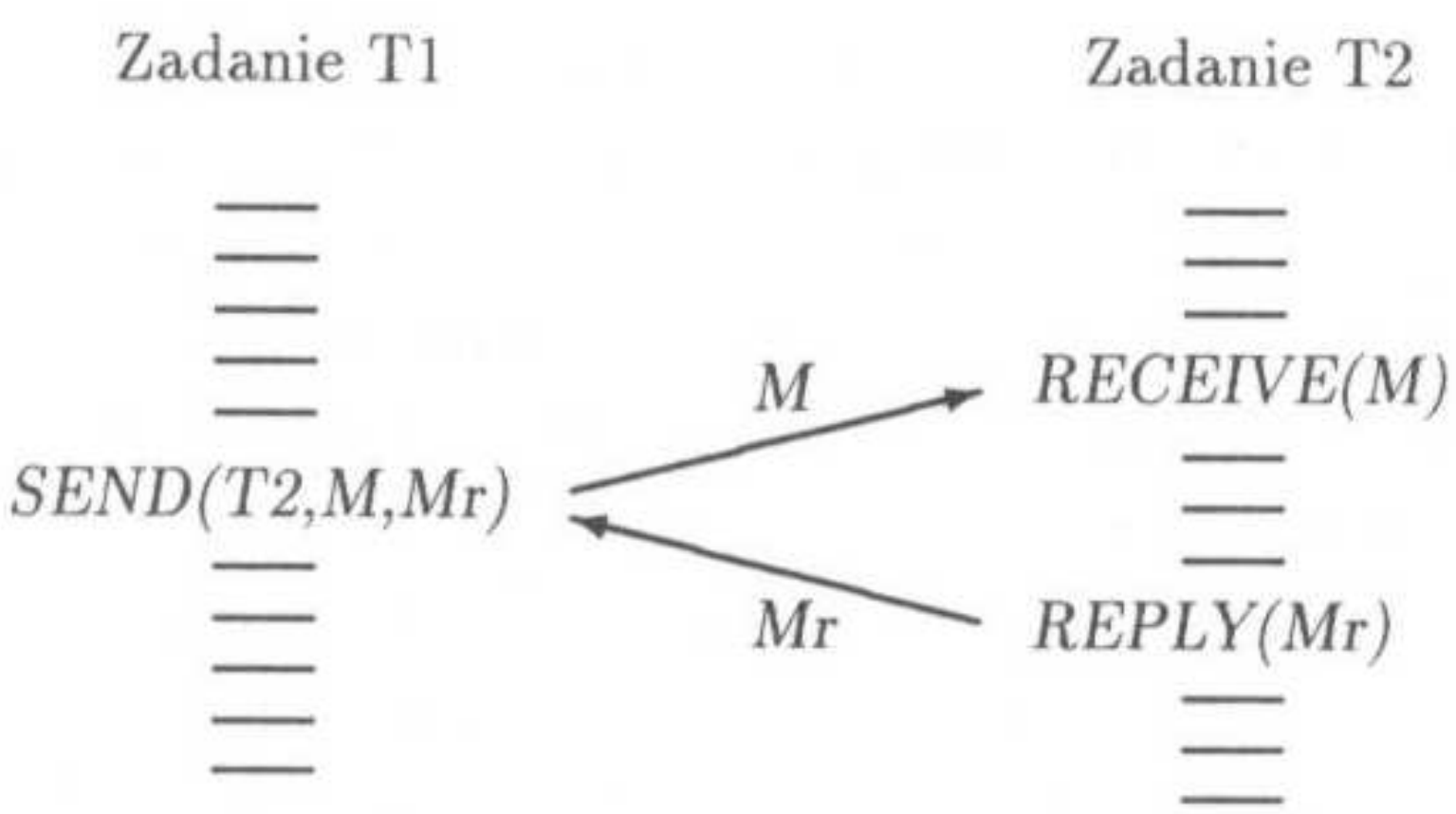
Mechanizmy komunikacji:

- synchroniczny przekaz (spotkanie) polega na przekazaniu pary: wiadomość –odpowiedź.
- Funkcja Send – powoduje wysłanie wiadomości do drugiego zadania i zawieszenie zadania nadającego do otrzymania odpowiedzi.
- Funkcja Receive – odbiera wiadomość
- Funkcja Reply – przekazuje odpowiedź i wznawia zadanie nadające.

Mechanizm ten umożliwia bezpośredni przekaz pomiędzy zadaniami bez konieczności buforowania wiadomości. Wymaga znajomości adresata i odbiorcy wiadomości. W przypadku wystąpienia żądania kilku spotkań z jednym zadaniem, następuje zawieszenie zadań i ustawienie kolejki zgodnie z algorytmem szeregującym.

System operacyjny QNX

QNX – komunikacja - spotkania



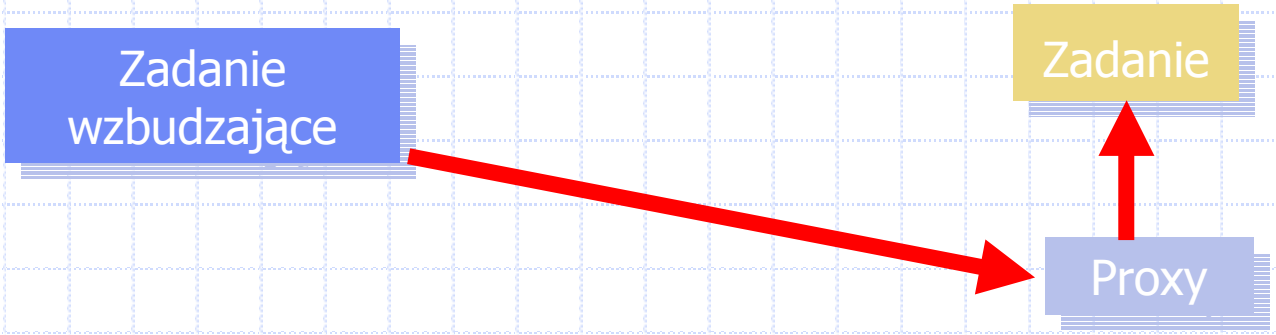
System operacyjny czasu rzeczywistego

QNX – komunikacja - proxy

Proxy to systemowe obiekty pośredniczące, przekazujące informacje i sygnały synchronizujące pomiędzy zadaniami.

Komunikacja polega na:
- przypisaniu obiektu proxy do zadania (właściciela),
- zapisie wiadomości w buforze wewnętrznym
- wzbudzeniu obiektu proxy (każde zadanie może wzbudzić proxy)
- obiekt proxy wysyła informację o wzbudzeniu do swego właściciela
- zadanie wzbudzenia obiektu proxy nie ulega zawieszeniu.

Komunikacja proxy jest stosowana głównie do jednokierunkowego przekazania sygnałów synchronizacyjnych.



System operacyjny QNX

QNX – komunikacja – sygnały i potoki

Sygnałami są informacje o błędach i stanach nienormalnych. Występuje 29 stałych sygnałów (naruszenie ochrony pamięci, nieuprawnione wykonanie instrukcji, itp.) oraz 2 definiowalne przez użytkownika. Sygnały mogą być generowane przez zadania użytkowe i systemowe. Otrzymanie sygnału polega na przerwanie toku przetwarzania i wykonanie podprogramu obsługi sygnału. Mechanizm obsługi sygnałów jest zbliżony do mechanizmu obsługi przerwań. Sygnały obsługiwane są wyłącznie lokalnie w ramach realizowanego zadania. Brak odpowiedniego programu obsługi sygnału kończy zadanie lub ignoruje sygnał.

Potoki to jednokierunkowe bufory komunikacyjne o strukturze FIFO. Potoki nie podlegają zawieszeniu w trakcie operacji zapisu lub odczytu. Przeznaczone są do przekazywanie danych między sekwencyjnie wykonywanymi zadaniami. Ze względu na stosukowo długi czas realizacji mają ograniczone zastosowania.

System operacyjny QNX

QNX – obsługa zdarzeń i uzależnienia czasowe

Przerwania są obsługiwane przez jądro systemu. Przerwania mają przyporządkowane różne priorytety. Przerwanie o wyższym priorytecie może przerwać zadanie obsługi przerwania o niższym priorytecie. W przypadku przypisania do jednego przerwania wielu zadań następuje przypadkowy wybór jednego z nich w momencie wystąpienia przerwania.

Uprzywilejowane zadania mogą mieć własne podprogramy.

Przerwanie można skierować na poziom zadania, podprogram może np. wzbudzić obiekt proxy, który prześle wiadomość dotyczącą przerwania.

QNX – informowanie zadań sterujących o zdarzeniach w urządzeniach zewnętrznych

Dwie podstawowe funkcje kontrolujące czas to:

- sleep() – 1s
- delay() – 1ms

W wielu przypadkach bardziej wygodne jest zastosowanie liczników czasu, inicjowanych przez zadania co: 0,50 .. 50ms

Utworzenie licznika wymaga określenie rodzaju operacji wykonanej po zakończeniu operacji zliczania czasu (wznowienie zadania, wzbudzenie obiektu proxy, wysłanie sygnału).

System operacyjny QNX

QNX – komunikacja w sieci - połączenia

QNX stosuje w komunikacji sieciowej model warstwowy zgodny z modelem referencyjnym otwartych systemów komunikacyjnych ISO/OSI:

Warstwa fizyczna obsługuje sieci: Arcnet i Ethernet

Warstwa sieciowa i transportowa obsługuje protokół TCP/IP, co daje możliwość komunikacji w sieciach heterogenicznych.

Spotkania, proxy i sygnały realizowane są zgodnie z protokołem połączeniowym poprzez połączenia znajdujące się pod nadzorem systemu.

System kontroluje: sprawność sprzętu komunikacyjnego (okresowe sygnały diagnostyczne) oraz poprawność transmisji. Jest odpowiedzialny za ewentualne retransmisje.

Połączenie (dwukierunkowe) inicjowane przez nadawcę, tworzy wirtualne zadania, które reprezentują lokalnie zadania odległe.

Zakończenie jednego zadania powoduje zakończenie wszystkich zadań wirtualnych i wysłanie informacji o zerwaniu połączenia.

System operacyjny QNX

QNX – komunikacja w sieci – nazwy zadań

QNX nadaje automatycznie każdemu zadaniu identyfikator numeryczny oraz kojarzy nazwę symboliczną zadania z numerem węzła i identyfikatorem zadania w węźle. Zadania obsługiwane są globalnie przez Administratora Nazw (zadanie systemowe).

Identyfikacja zadania polega na wysłaniu zapytania do Administratora w postaci nazwy symbolicznej i odebraniu identyfikatora zadania zapisanego w katalogu zadań. Zadanie wykonywane w innym węźle powoduje automatyczne otwarcie połączenia.

Zadania mogą być również zdefiniowane lokalnie dla danego węzła. Ich nazwy są również przechowywane przez Administratora Nazw w specjalnym katalogu.